

CERTIFIED COPY OF  
PRIORITY DOCUMENT

本 国 特 許 庁  
PATENT OFFICE  
JAPANESE GOVERNMENT

OLD  
11000 U.S. PTO  
09/866288  
06/25/01

別紙添付の書類に記載されている事項は下記の出願書類に記載されて  
いる事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed  
with this Office.

出 願 年 月 日

Date of Application:

2000年 5月29日

出 願 番 号

Application Number:

特願2000-158546

出 願 人

Applicant(s):

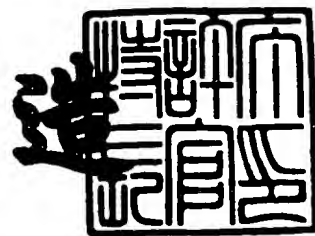
ソニー株式会社

Best Available Copy

2001年 4月13日

特許庁長官  
Commissioner,  
Patent Office

及 川 耕 造



出証番号 出証特2001-3030657

【書類名】 特許願

【整理番号】 0000216606

【提出日】 平成12年 5月29日

【あて先】 特許庁長官殿

【国際特許分類】 H04N 1/64

【発明者】

    【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社  
                                内

    【氏名】 上田 衛

【特許出願人】

    【識別番号】 000002185

    【氏名又は名称】 ソニー株式会社

    【代表者】 出井 伸之

【代理人】

    【識別番号】 100082131

    【弁理士】

    【氏名又は名称】 稲本 義雄

    【電話番号】 03-3369-6479

【手数料の表示】

    【予納台帳番号】 032089

    【納付金額】 21,000円

【提出物件の目録】

    【物件名】 明細書 1

    【物件名】 図面 1

    【物件名】 要約書 1

    【包括委任状番号】 9708842

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 復号装置および方法、並びに記録媒体

【特許請求の範囲】

【請求項 1】 符号化ストリームを復号する復号装置において、  
高速化された前記符号化ストリームを入力する入力手段と、  
前記高速化された前記符号化ストリームを復号する複数の復号手段と、  
複数の前記復号手段を並行して動作させるように制御する復号制御手段と、  
前記複数の復号手段が復号した前記高速化された前記符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御手段と  
を含むことを特徴とする復号装置。

【請求項 2】 前記高速化された前記符号化ストリームは、ビットレートが  
所定数倍に高速化された MPEG 2 ビデオビットストリームである  
ことを特徴とする請求項 1 に記載の復号装置。

【請求項 3】 前記出力制御手段は、ビットレートが前記所定数倍に高速化  
された前記 MPEG 2 ビデオビットストリームに対応する画像を、0 倍乃至前記所定  
数倍の再生速度で出力させる  
ことを特徴とする請求項 2 に記載の復号装置。

【請求項 4】 前記復号手段は、復号処理の終了を示す信号を前記復号制御  
手段に出力し、  
前記復号制御手段は、復号処理の終了を示す前記信号を出力した前記復号手段  
に、新たな前記符号化ストリームを復号させるように制御する  
ことを特徴とする請求項 1 に記載の復号装置。

【請求項 5】 前記符号化ストリームをバッファリングする第 1 のバッファ  
手段と、  
前記符号化ストリームから、前記符号化ストリームに含まれる所定の情報の単  
位の始まりを表わすスタートコードを読み出すとともに、前記第 1 のバッファ手  
段に、前記スタートコードが保持されている位置に関する位置情報を読み出す読  
み出し手段と、

前記読み出し手段により読み出された、前記スタートコードおよび前記位置情

報をバッファリングする第 2 のバッファ手段と、

前記第 1 のバッファ手段による前記符号化ストリームのバッファリング、および前記第 2 のバッファ手段による前記スタートコードおよび前記位置情報のバッファリングを制御するバッファリング制御手段と

をさらに含むことを特徴とする請求項 1 に記載の復号装置。

【請求項 6】 複数の前記復号手段により復号され、出力された複数の画像データのうちの所定のものを選択する選択手段と、

前記選択手段により選択された前記画像データの入力を受け、必要に応じて動き補償を施す動き補償手段と

をさらに含むことを特徴とする請求項 1 に記載の復号装置。

【請求項 7】 前記復号手段は、復号処理が終了したことを示す終了信号を前記選択手段に出力し、

前記選択手段は、

複数の前記復号手段のそれぞれの処理状態に対応する値を記憶する記憶手段を含み、

前記記憶手段が記憶した値が全て第 1 の値になった場合、復号処理が終了したことを示す前記終了信号を出力している前記復号手段に対応する前記記憶手段に記憶されている値を、前記第 1 の値から第 2 の値に変更し、

対応する前記記憶手段に記憶されている値が前記第 2 の値である前記第 1 の復号手段により復号された前記画像データうち、いずれかの前記画像データを選択し、

選択された前記画像データを復号した前記復号手段に対応する前記記憶手段に記憶されている値を前記第 1 の値に変更する

ことを特徴とする請求項 6 に記載の復号装置。

【請求項 8】 前記選択手段により選択された前記画像データ、または前記動き補償手段により動き補償が施された前記画像データを保持する保持手段と、

前記選択手段により選択された前記画像データ、または前記動き補償手段により動き補償が施された前記画像データの前記保持手段による保持を制御する保持制御手段と

をさらに含むことを特徴とする請求項 6 に記載の復号装置。

【請求項 9】 前記保持手段は、前記画像データの輝度成分と色差成分をそれぞれ分けて保持する

ことを特徴とする請求項 8 に記載の復号装置。

【請求項 10】 符号化ストリームを復号する復号装置の復号方法において

、  
高速化された前記符号化ストリームを入力する入力ステップと、  
前記高速化された前記符号化ストリームを復号する複数の復号ステップと、  
複数の前記復号ステップの処理を並行して動作させるように制御する復号制御ステップと、

前記複数の復号ステップの処理で復号された前記高速化された前記符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップと  
を含むことを特徴とする復号方法。

【請求項 11】 符号化ストリームを復号する復号用のプログラムであって

、  
高速化された前記符号化ストリームを入力する入力ステップと、  
前記高速化された前記符号化ストリームを復号する複数の復号ステップと、  
複数の前記復号ステップの処理を並行して動作させるように制御する復号制御ステップと、

前記複数の復号ステップの処理で復号された前記高速化された前記符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップと

を含むことを特徴とするコンピュータが読み取り可能なプログラムが記録されている記録媒体。

【請求項 12】 符号化ストリームを復号する複数のスライスデコーダを備える復号装置において、

高速化された前記符号化ストリームを入力する入力手段と、  
複数の前記スライスデコーダを並行して動作させるように制御するスライスデコーダ制御手段と、

前記複数のスライスデコーダが復号した前記高速化された前記符号化ストリー

ムに対応する画像を任意の再生速度で出力させる出力制御手段と  
を含むことを特徴とする復号装置。

【請求項 1 3】 符号化ストリームを復号する複数のスライスデコーダを備える復号装置の復号方法において、

高速化された前記符号化ストリームを入力する入力ステップと、

複数の前記スライスデコーダを並行して動作させるように制御するスライスデコーダ制御ステップと、

前記複数のスライスデコーダが復号した前記高速化された前記符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップと  
を含むことを特徴とする復号方法。

【請求項 1 4】 符号化ストリームを復号する複数のスライスデコーダを備える復号装置の復号用のプログラムであって、

高速化された前記符号化ストリームを入力する入力ステップと、

複数の前記スライスデコーダを並行して動作させるように制御するスライスデコーダ制御ステップと、

前記複数のスライスデコーダが復号した前記高速化された前記符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップと

を含むことを特徴とするコンピュータが読み取り可能なプログラムが記録されている記録媒体。

#### 【発明の詳細な説明】

##### 【 0 0 0 1 】

#### 【発明の属する技術分野】

本発明は、復号装置および方法、並びに記録媒体に関し、例えば、MPEG 2 ビデオビットストリームをデコードする場合に用いて好適な復号装置および方法、並びに記録媒体に関する。

##### 【 0 0 0 2 】

#### 【従来の技術】

MPEG(Moving Picture Experts Group) 2 ビデオは、ISO/IEC(International Standards Organization/International Electrotechnical Commission)13818-2、

およびITU-T(International Telecommunication Union-Telecommunication sector)勧告H.262に規定されているビデオ信号の高能率圧縮符号化方式である。

### 【 0 0 0 3 】

MPEG 2 ビデオでは、ビデオ画像の各画像は符号化の効率が異なる 3 つのピクチャタイプ (フレーム内符号化画像 (I (Intra)ピクチャ)、フレーム間順方向予測符号化画像 (P (Predictive)ピクチャ)、および双方向予測符号化画像 (B (Bidirectionally predictive)ピクチャ)) のうちのいずれかに分類される。I ピクチャに分類された画像は、当該画像のフレーム内の空間的相関関係に基づいて符号化される。P ピクチャに分類された画像は、当該画像の前に存在する I ピクチャまたは P ピクチャからの動き補償予測によって符号化される。B ピクチャに分類された画像は、当該画像の前後に存在する I ピクチャまたは P ピクチャからの動き補償予測によって符号化される。したがって、符号化の効率は、I ピクチャ、P ピクチャ、B ピクチャの順に高くなる。

### 【 0 0 0 4 】

図 1 を参照して具体的に説明する。ビデオ信号の画像が  $I_1$ ,  $B_2$ ,  $B_3$ ,  $P_4$ ,  $\dots$ ,  $P_{13}$  に分類された場合 (下付の数字は表示順序を示す)、例えば、画像  $I_1$  はフレーム内の空間的相関関係に基づいて符号化され、画像  $P_4$  は、画像  $I_1$  からの動き補償予測によって符号化され、画像  $P_7$  は、画像  $P_4$  からの動き補償予測によって符号化される。例えばまた、画像  $B_2$  は、画像  $I_1$  および画像  $P_4$  からの動き補償予測によって符号化され、画像  $B_5$  は、画像  $P_4$  および画像  $P_7$  からの動き補償予測によって符号化される。

### 【 0 0 0 5 】

MPEG 2 符号化ストリームは、符号化の手法によって決まるプロファイルと、取り扱う画素数によって決まるレベルによってクラス分けされ、広範囲なアプリケーションに対応できるようになされている。例えば、MPEG 2 符号化ストリームのクラスのうちの 1 つである MP@ML (メイン・プロファイル・メイン・レベル) は、DVB(Digital Video Broadcast)や、DVD(Digital Versatile Disk)に広く実用化されている。プロファイルおよびレベルは、図 6 を用いて後述する sequence\_extension に記述される。

## 【 0 0 0 6 】

また、放送局における用途に適用させたMPEG 2 符号化ストリームのプロファイルとして、ビデオの色差信号を従来のベースバンドと同様に4:2:2方式で取り扱うことができるようにビットレートの上限を高く設定した4:2:2P (4:2:2プロファイル) が規定されている。さらに、MPEG 2 符号化ストリームのレベルとして、次世代の高解像度ビデオ信号に対応するHL (ハイ・レベル) が規定されている。

## 【 0 0 0 7 】

図2は、MPEG 2 で規定されている代表的なクラスである、4:2:2P@HL (4:2:2プロファイル・ハイ・レベル)、4:2:2P@ML (4:2:2プロファイル・メイン・レベル)、MP@HL (メイン・プロファイル・ハイ・レベル)、MP@HL-1440 (メイン・プロファイル・ハイ・レベル-1440)、MP@ML (メイン・プロファイル・メイン・レベル)、MP@LL (メイン・プロファイル・ロー・レベル)、および、SP@ML (シンプル・プロファイル・メイン・レベル) について、各クラスのパラメータ (ビットレート、1ラインあたりのサンプル数、1フレームあたりのライン数、フレームの処理時間、およびサンプルの処理時間) の上限値がそれぞれ示されている。

## 【 0 0 0 8 】

図2に示すように、4:2:2P@HLのビットレートの上限値は、300 (メガビット/秒) であり、処理する画素数の上限値は、62,668,800 (画素/秒) である。一方、MP@MLのビットレートの上限値は、15 (メガビット/秒) であり、処理する画素数の上限値は、10,368,000 (画素/秒) である。すなわち、4:2:2P@HLをデコードするビデオデコーダは、MP@MLをデコードするビデオデコーダに比較して、ビットレートは20倍、処理する画素数は約6倍の処理能力が必要であることがわかる。

## 【 0 0 0 9 】

ここで、MPEG 2 ビデオビットストリームのレベル構造について図3を参照して説明する。最上位層であるピクチャ層の最初には、sequence\_headerが記述されている。sequence\_headerは、MPEGビットストリームのシーケンスのヘッダデー



タを定義するものである。シーケンス最初のsequence\_headerに、sequence\_extensionが続かない場合、当該ビットストリームには、ISO/IEC11172-2の規定が適応される。シーケンスの最初のsequence\_headerに、sequence\_extensionが続く場合、その後に発生する全てのsequence\_headerの直後には、sequence\_extensionが続く。すなわち、図3に示す場合においては、全てのsequence\_headerの直後に、sequence\_extensionが続く。

## 【 0 0 1 0 】

sequence\_extensionは、MPEGビットストリームのシーケンス層の拡張データを定義するものである。sequence\_extensionは、sequence\_headerの直後にのみ発生し、かつ、復号後、およびフレームリオーダリング後にフレームの損失がないようにするために、ビットストリームの最後に位置するsequence\_end\_codeの直前にきてはならない。また、ビットストリーム中に、sequence\_extensionが発生した場合、それぞれのpicture\_headerの直後にpicture\_coding\_extensionが続く。

## 【 0 0 1 1 】

GOP(Group Of Picture)内には、複数の画像(picture)が含まれる。GOP\_headerは、MPEGビットストリームのGOP層のヘッダデータを定義するものであり、さらに、このビットストリーム中には、picture\_headerとpicture\_coding\_extensionによって定義されたデータエレメントが記述されている。1枚の画像は、picture\_headerおよびpicture\_coding\_extensionに続くpicture\_dataとして符号化される。また、GOP\_headerに続く最初の符号化フレームは、符号化されたIフレームである（すなわち、GOP\_headerの最初の画像はIピクチャである）。ITU-T勧告H.262には、sequence\_extensionおよびpicture\_coding\_extensionの他、各種の拡張が定義されているが、ここでは図示、および説明は省略する。

## 【 0 0 1 2 】

picture\_headerは、MPEGビットストリームのピクチャ層のヘッダデータを定義するものであり、picture\_coding\_extensionは、MPEGビットストリームのピクチャ層の拡張データを定義するものである。

## 【 0 0 1 3 】

picture\_dataは、MPEGビットストリームのスライス層およびマクロブロック層に関するデータエレメントを記述するものである。picture\_dataは、図3に示されるように、複数のslice（スライス）に分割され、スライスは、複数のmacro\_block（マクロブロック）に分割される。

#### 【0014】

マクロブロックは、 $16 \times 16$ の画素データで構成されている。スライスの最初のマクロブロックおよび最後のマクロブロックは、スキップマクロブロック（情報を含まないマクロブロック）ではない。なお、フレームDCT（Discrete Cosine Transform：離散コサイン変換）符号化およびフィールドDCT符号化の使用が可能なフレーム画像においては、フレーム符号化が使用されたマクロブロックの内部構成とフィールド符号化が使用されたマクロブロックの内部構成が相違する。

#### 【0015】

マクロブロックは、輝度成分および色差成分の1区画を含む。マクロブロックという用語は、情報源および復号データまたは対応する符号化データ成分のいずれかを示す。マクロブロックには、4:2:0、4:2:2および4:4:4の3つの色差フォーマットがある。マクロブロックにおけるブロックの順序は、それぞれの色差フォーマットによって異なる。

#### 【0016】

図4（A）に、色差フォーマットが4:2:0方式である場合におけるマクロブロックを示す。4:2:0方式の場合、マクロブロックは、4個の輝度（Y）ブロックと、それぞれ1個の色差（Cb, Cr）ブロックで構成される。図4（B）に、色差フォーマットが4:2:2方式である場合におけるマクロブロックを示す。4:2:2方式の場合、マクロブロックは、4個の輝度（Y）ブロックと、それぞれ2個の色差（Cb, Cr）ブロックで構成される。

#### 【0017】

各マクロブロックは、いくつかの方法により、予測符号化処理が可能である。予測モードは、フィールド予測とフレーム予測の2種類に大別される。フィールド予測においては、先に復号された、1つ、もしくは複数のフィールドのデータ

を使用し、各フィールドについて、独自に予測を行う。フレーム予測は、先に復号された、1つ、もしくは複数のフレームを使用してフレームの予測を行う。フィールド画像内では、予測は全てフィールド予測である。一方、フレーム画像においては、フィールド予測、またはフレーム予測のいずれかにより予測が可能であり、その予測方法は、マクロブロックごとに選択される。また、マクロブロックの予測符号化処理においては、フィールド予測およびフレーム予測以外に、16×8動き補償およびデュアルプライムの2種類の特別予測モードを使用することができる。

## 【0018】

動きベクトル情報、および他の周辺情報は、各マクロブロックの予測誤差信号とともに符号化される。動きベクトルの符号化については、可変長符号を使用して符号化された最後の動きベクトルを予測ベクトルとして、予測ベクトルとの差分ベクトルを符号化する。表示可能なベクトルの最大長は、画像毎にプログラムすることができる。また、適切な動きベクトルの計算は符号器により実行される。

## 【0019】

picture\_dataの後には、次のsequence\_headerとsequence\_extensionが配置されている。このsequence\_headerとsequence\_extensionによって記述されたデータエレメントは、ビデオストリームのシーケンスの先頭に記述されたsequence\_headerとsequence\_extensionによって記述されたデータエレメントと全く同じである。このように、同じデータをストリーム中に記述することにより、ビットストリーム受信装置側において、データストリームの途中（例えばピクチャ層に対応するビットストリーム部分）から受信が開始された場合、シーケンス層のデータを受信できなくなってストリームをデコードできなくなることが抑止される。

## 【0020】

最後のsequence\_headerとsequence\_extensionとによって定義されたデータエレメントの次、つまり、データストリームの最後には、シーケンスの終わりを示す32ビットのsequence\_end\_codeが記述されている。

## 【0021】

次に、図 5 乃至 1 2 を用いて、それぞれのデータ要素の詳細について説明する。

#### 【 0 0 2 2 】

図 5 に、sequence\_header のデータ構成を示す。sequence\_header に含まれるデータ要素は、sequence\_header\_code、horizontal\_size\_value、vertical\_size\_value、aspect\_ratio\_information、frame\_rate\_code、bit\_rate\_value、marker\_bit、vbv\_buffer\_size\_value、constrained\_parameter\_flag、load\_intra\_quantiser\_matrix、intra\_quantiser\_matrix、load\_non\_intra\_quantiser\_matrix、およびnon\_intra\_quantiser\_matrix等から構成される。

#### 【 0 0 2 3 】

sequence\_header\_code は、シーケンス層のスタート同期コードを表すデータである。horizontal\_size\_value は、画像の水平方向の画素数の下位 1 2 ビットからなるデータである。vertical\_size\_value は、画像の縦のライン数の下位 1 2 ビットからなるデータである。aspect\_ratio\_information は、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame\_rate\_code は、画像の表示周期を表すデータである。bit\_rate\_value は、発生ビット量に対する制限のためのビットレートの下位 1 8 ビットのデータである。

#### 【 0 0 2 4 】

marker\_bit は、スタートコードエミュレーションを防止するために挿入されるビットデータである。vbv\_buffer\_size\_value は、発生符号量制御用の仮想バッファ VBV (Video Buffering Verifier) の大きさを決める値の下位 1 0 ビットデータである。constrained\_parameter\_flag は、各パラメータが制限以内であることを示すデータである。load\_non\_intra\_quantiser\_matrix は、非イントラマクロブロック用量子化マトリックス・データの存在を示すデータである。load\_intra\_quantiser\_matrix は、イントラマクロブロック用量子化マトリックス・データの存在を示すデータである。intra\_quantiser\_matrix は、イントラマクロブロック用量子化マトリックスの値を示すデータである。non\_intra\_quantiser\_matrix は、非イントラマクロブロック用量子化マトリックスの値を表すデータである。

#### 【 0 0 2 5 】

図 6 に、sequence\_extension のデータ構成を示す。sequence\_extension は、extension\_start\_code、extension\_start\_code\_identifier、profile\_and\_level\_indication、progressive\_sequence、chroma\_format、horizontal\_size\_extension、vertical\_size\_extension、bit\_rate\_extension、marker\_bit、vbv\_buffer\_size\_extension、low\_delay、frame\_rate\_extension\_n、および frame\_rate\_extension\_d 等のデータエレメントから構成されている。

#### 【 0 0 2 6 】

extension\_start\_code は、エクステンションデータ（拡張データ）のスタート同期コードを表すデータである。extension\_start\_code\_identifier は、拡張データの種類を示すデータである。profile\_and\_level\_indication は、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive\_sequence は、ビデオデータが順次走査（プログレッシブ画像）であることを示すデータである。chroma\_format は、ビデオデータの色差フォーマットを指定するためのデータである。horizontal\_size\_extension は、シーケンスヘッダの horizontal\_size\_value に加える上位 2 ビットのデータである。vertical\_size\_extension は、シーケンスヘッダの vertical\_size\_value に加える上位 2 ビットのデータである。

#### 【 0 0 2 7 】

bit\_rate\_extension は、シーケンスヘッダの bit\_rate\_value に加える上位 1 2 ビットのデータである。marker\_bit は、スタートコードエミュレーションを防止するために挿入されるビットデータである。vbv\_buffer\_size\_extension は、シーケンスヘッダの vbv\_buffer\_size\_value に加える上位 8 ビットのデータである。low\_delay は、B ピクチャを含まないことを示すデータである。frame\_rate\_extension\_n は、シーケンスヘッダの frame\_rate\_code と組み合わせてフレームレートを得るためのデータである。frame\_rate\_extension\_d は、シーケンスヘッダの frame\_rate\_code と組み合わせてフレームレートを得るためのデータである。

#### 【 0 0 2 8 】

図 7 に、GOP\_header のデータ構成を示す。GOP\_header を表わすデータエレメントは、group\_start\_code、time\_code、closed\_gop、および broken\_link から構成

される。

#### 【 0 0 2 9 】

group\_start\_codeは、GOP層の開始同期コードを示すデータである。time\_codeは、GOPの先頭ピクチャの時間を示すタイムコードである。closed\_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken\_linkは、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

#### 【 0 0 3 0 】

図8に、picture\_headerのデータ構成を示す。picture\_headerに関するデータエレメントは、picture\_start\_code、temporal\_reference、picture\_coding\_type、vbv\_delay、full\_pel\_forward\_vector、forward\_f\_code、full\_pel\_backward\_vector、およびbackward\_f\_code等から構成される。

#### 【 0 0 3 1 】

picture\_start\_codeは、ピクチャ層の開始同期コードを表すデータである。temporal\_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture\_coding\_typeは、ピクチャタイプを示すデータである。vbv\_delayは、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full\_pel\_forward\_vector、forward\_f\_code、full\_pel\_backward\_vector、およびbackward\_f\_codeは、MPEG2では使用されない固定データである。

#### 【 0 0 3 2 】

図9に、picture\_coding\_extensionのデータ構成を示す。picture\_coding\_extensionは、extension\_start\_code、extension\_start\_code\_identifier、f\_code[0][0]、f\_code[0][1]、f\_code[1][0]、f\_code[1][1]、intra\_dc\_precision、picture\_structure、top\_field\_first、frame\_pred\_frame\_dct、concealment\_motion\_vectors、q\_scale\_type、intra\_vlc\_format、alternate\_scan、repeat\_first\_field、chroma\_420\_type、progressive\_frame、composite\_display\_flag、v\_axis、field\_sequence、sub\_carrier、burst\_amplitude、およびsub\_carrier\_phase等から構成される。

#### 【 0 0 3 3 】

`extension_start_code`は、ピクチャ層のエクステンションデータ（拡張データ）のスタートを示す同期コードを表すデータである。`extension_start_code_identifier`は、拡張データの種類を示すデータである。`f_code[0][0]`は、フォワード方向の水平動きベクトル探索範囲を表すデータである。`f_code[0][1]`は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。`f_code[1][0]`は、バックワード方向の水平動きベクトル探索範囲を表すデータである。`f_code[1][1]`は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。

## 【 0 0 3 4 】

`intra_dc_precision`は、DC係数の精度を表すデータである。ブロック内の各画素の輝度および色差信号を表した行列  $f$  にDCTを施すと、 $8 \times 8$ のDCT係数行列  $F$  が得られる。DCT係数行列  $F$  の左上隅の係数をDC係数と呼ぶ。DC係数はブロック内の平均輝度、平均色差を表わす信号である。`picture_structure`は、フレームストラクチャであるか、フィールドストラクチャであるかを示すデータであり、フィールドストラクチャであることを示している場合はさらに、上位フィールドであるか、下位フィールドであることを示すデータを含んでいる。`top_field_first`は、フレームストラクチャである場合において、最初のフィールドが上位であるか、下位であることを示すデータである。`frame_predictive_frame_dct`は、フレームストラクチャである場合において、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。`concealment_motion_vectors`は、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがっていることを示すデータである。

## 【 0 0 3 5 】

`q_scale_type`は、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。`intra_vlc_format`は、イントラマクロブロックに、別の2次元VLC(Variable Length Coding)を使うか否かを示すデータである。`alternate_scan`は、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。`repeat_first_field`は、2:3プルダウンの際に使われるデータである。`chroma_420_type`には、色差フォーマットが4:2:0方式である場合、次の`progressive_frame`と同じ値が記述され、色差フォーマットが

4:2:0方式ではない場合には0が記述される。progressive\_frameは、このピクチャが順次走査であるか否かを示すデータである。composite\_display\_flagは、ソース信号がコンポジット信号であったか否かを示すデータである。v\_axis、field\_sequence、sub\_carrier、burst\_amplitude、およびsub\_carrier\_phaseは、ソース信号がコンポジット信号であった場合に使われるデータである。

## 【 0 0 3 6 】

図10に、picture\_dataのデータ構成を示す。picture\_data()関数によって定義されるデータエレメントは、slice()関数によって定義されるデータエレメントである。このslice()関数によって定義されるデータエレメントは、ビットストリーム中に少なくとも1個記述されている。

## 【 0 0 3 7 】

slice()関数は、図11に示されるように、slice\_start\_code、quantiser\_scale\_code、intra\_slice\_flag、intra\_slice、reserved\_bits、extra\_bit\_slice、およびextra\_information\_slice等のデータエレメントと、macroblock()関数によって定義される。

## 【 0 0 3 8 】

slice\_start\_codeは、slice()関数によって定義されるデータエレメントのスタートを示すスタートコードである。quantiser\_scale\_codeは、このスライス層に存在するマクロブロックに対して設定された量子化ステップサイズを示すデータであるが、マクロブロック毎に、quantiser\_scale\_codeが設定されている場合には、各マクロブロックに対して設定されたmacroblock\_quantiser\_scale\_codeのデータが優先して使用される。

## 【 0 0 3 9 】

intra\_slice\_flagは、ビットストリーム中にintra\_sliceおよびreserved\_bitsが存在するか否かを示すフラグである。intra\_sliceは、スライス層中にノンイントラマクロブロックが存在するか否かを示すデータである。スライス層におけるマクロブロックのいずれかがノンイントラマクロブロックである場合には、intra\_sliceは「0」となり、スライス層におけるマクロブロックの全てがノンイントラマクロブロックである場合には、intra\_sliceは「1」となる。reserved\_



bitsは、7ビットの予備のデータ領域である。extra\_bit\_sliceは、追加の情報が存在するか否かを示すフラグであって、次にextra\_information\_sliceが存在する場合には「1」に設定され、追加の情報が存在しない場合には「0」に設定される。

#### 【0 0 4 0】

これらのデータ要素の次には、macroblock()関数によって定義されたデータ要素が記述されている。macroblock()関数は、図12に示すように、macroblock\_escape、macroblock\_address\_increment、およびquantiser\_scale\_code、およびmarker\_bit等のデータ要素と、macroblock\_modes()関数、motion\_vectors()関数、およびcoded\_block\_pattern()関数によって定義されたデータ要素を記述するための関数である。

#### 【0 0 4 1】

macroblock\_escapeは、参照マクロブロックと前のマクロブロックとの水平方向の差が34以上であるか否かを示す固定ビット列である。参照マクロブロックと前のマクロブロックとの水平方向の差が34以上である場合、macroblock\_address\_incrementの値に33が加えられる。macroblock\_address\_incrementは、参照マクロブロックと前のマクロブロックとの水平方向の差を示すデータである。もし、macroblock\_address\_incrementの前にmacroblock\_escapeが1つ存在するのであれば、このmacroblock\_address\_incrementの値に33を加えた値が、実際の参照マクロブロックと前のマクロブロックとの水平方向の差分を示すデータとなる。

#### 【0 0 4 2】

quantiser\_scale\_codeは、各マクロブロックに設定された量子化ステップサイズを示すデータであり、macroblock\_quantが「1」のときだけ存在する。各スライス層には、スライス層の量子化ステップサイズを示すslice\_quantiser\_scale\_codeが設定されているが、参照マクロブロックに対してscale\_codeが設定されている場合には、この量子化ステップサイズを選択する。

#### 【0 0 4 3】

macroblock\_address\_incrementの次には、macroblock\_modes()関数によって定

義されるデータエレメントが記述されている。macroblock\_modes()関数は、図13に示すように、macroblock\_type、frame\_motion\_type、field\_motion\_type、dct\_type等のデータエレメントを記述するための関数である。macroblock\_typeは、マクロブロックの符号化タイプを示すデータである。

## 【0044】

macroblock\_motion\_forwardまたはmacroblock\_motion\_backwardが「1」であり、ピクチャ構造がフレームであり、さらにframe\_pred\_frame\_dctが「0」である場合、macroblock\_typeを表わすデータエレメントの次にframe\_motion\_typeを表わすデータエレメントが記述される。なお、このframe\_pred\_frame\_dctは、frame\_motion\_typeがビットストリーム中に存在するか否かを示すフラグである。

## 【0045】

frame\_motion\_typeは、フレームのマクロブロックの予測タイプを示す2ビットのコードである。予測ベクトルが2個であって、フィールドベースの予測タイプである場合、frame\_motion\_typeには「00」が記述される。予測ベクトルが1個であって、フィールドベースの予測タイプである場合、frame\_motion\_typeには「01」が記述される。予測ベクトルが1個であって、フレームベースの予測タイプである場合、frame\_motion\_typeには「10」が記述される。予測ベクトルが1個であって、デュアルプライムの予測タイプである場合、frame\_motion\_typeには「11」が記述される。

## 【0046】

field\_motion\_typeは、フィールドのマクロブロックの動き予測を示す2ビットのコードである。予測ベクトルが1個であって、フィールドベースの予測タイプである場合、field\_motion\_typeには「01」が記述される。予測ベクトルが2個であって、18×8マクロブロックベースの予測タイプである場合、field\_motion\_typeには「10」が記述される。予測ベクトルが1個であって、デュアルプライムの予測タイプである場合、field\_motion\_typeには「11」が記述される。

## 【0047】

ピクチャ構造がフレームであり、frame\_pred\_frame\_dctが、そのビットストリ

ーム中にframe\_motion\_typeが存在することを示し、frame\_pred\_frame\_dctが、そのビットストリーム中にdct\_typeが存在することを示している場合、macroblock\_typeを表わすデータエレメントの次にはdct\_typeを表わすデータエレメントが記述される。なお、dct\_typeは、DCTがフレームDCTモードであるか、フィールドDCTモードであることを示すデータである。

## 【 0 0 4 8 】

MPEG 2 ビデオビットストリーム中の各データエレメントは、start codeと称される、特殊なビットパターンで開始される。これらのスタートコードは、別の状況では、ビデオストリーム中に現れない特定のビットパターンである。各スタートコードは、スタートコードプレフィクスと、それに続くスタートコード値から構成される。スタートコードプレフィクスは、ビット列“0000 0000 0000 0000 0000 0001”である。スタートコード値は、スタートコードのタイプを識別する 8 ビットのデータである。

## 【 0 0 4 9 】

図 1 4 に、MPEG 2 のスタートコード値(Start code value)を示す。多くのスタートコードには、1 個のスタートコード値が設定されている。しかしながら、slice\_start\_codeには、複数のスタートコード値(0 1 乃至 A F)が設定されており、このスタートコード値は、スライスに対する垂直位置を表わす。これらのスタートコードは、全てバイト単位であるため、スタートコードプレフィクスの最初のビットがバイトの最初のビットになるように、スタートコードプレフィクスの前に、複数のビット“0”が挿入され、スタートコードがバイト単位になるように調整される。

## 【 0 0 5 0 】

次に、MP@MLのMPEG 2 ビデオビットストリームに対応した従来のMPEGビデオデコーダについて、図 1 5 を参照して説明する。図 1 5 は、当該MPEGビデオデコーダの構成の一例を示している。

## 【 0 0 5 1 】

当該MPEGビデオデコーダは、ストリーム入力回路 1 1、バッファ制御回路 1 2、クロック発生回路 1 3、スタートコード検出回路 1 4、デコーダ 1 5、動き補

償回路 1 6、および表示出力回路 1 7 から構成される IC (Integrated Circuit) 1 と、ストリームバッファ 2 1 およびビデオバッファ 2 2 で構成され、例えば、DRAM (Dynamic Random Access Memory) からなるバッファ 2 により構成される。

#### 【 0 0 5 2 】

IC 1 のストリーム入力回路 1 1 は、高能率符号化された符号化ストリーム (MP@ML の MPEG 2 ビデオビットストリーム) の入力を受け、バッファ制御回路 1 2 に供給する。バッファ制御回路 1 2 は、クロック発生回路 1 3 から供給される基本クロックに従って、入力された符号化ストリームをバッファ 2 のストリームバッファ 2 1 に入力する。ストリームバッファ 2 1 は、少なくとも、MP@ML のデコードに要求される VBV バッファサイズである 1,835,008 ビットの容量を有する。ストリームバッファ 2 1 に保存されている符号化ストリームは、バッファ制御回路 1 2 の制御に従って、先に書き込まれたデータから順に読み出され、スタートコード検出回路 1 4 に供給される。スタートコード検出回路 1 4 は、入力されたストリームから、図 1 4 を用いて説明したスタートコードを検出し、検出したスタートコードおよび入力されたストリームをデコーダ 1 5 に出力する。

#### 【 0 0 5 3 】

デコーダ 1 5 は、入力されたストリームを MPEG シンタックスに基づいて、デコードする。デコーダ 1 5 は、入力されたスタートコードに従って、まず、ピクチャ層のヘッダパラメータをデコードし、それを基に、スライス層をマクロブロックに分離してマクロブロックをデコードし、その結果得られる予測ベクトルおよび画素を、動き補償回路 1 6 に出力する。

#### 【 0 0 5 4 】

圧縮符号化方式としての MPEG では、隣接した画像間の時間的冗長性を利用して、近接した画像間で動き補償した差分を得ることにより、符号化効率を改善している。当該 MPEG ビデオデコーダでは、動き補償を用いた画素に対しては、現在デコードしている画素にその動きベクトルが示す参照画像の画素データを加算することにより動き補償を行い、符号化前の画像データに復号する。

#### 【 0 0 5 5 】

デコーダ 1 5 から出力されるマクロブロックが動き補償を使用していない場合

、動き補償回路 1 6 は、その画素データをバッファ制御回路 1 2 を介してバッファ 2 のビデオバッファ 2 2 に書き込み、表示出力に備えるとともに、この画素データが、他の画像の参照データとされる場合に備える。

【 0 0 5 6 】

デコーダ 1 5 から出力されるマクロブロックが動き補償を使用している場合、動き補償回路 1 6 は、デコーダ 1 5 から出力される予測ベクトルに従って、バッファ制御回路 1 2 を介して、バッファ 2 のビデオバッファ 2 2 から参照画素データを読み出す。そして、読み出した参照画素データを、デコーダ 1 5 から供給された画素データに加算し、動き補償を行う。動き補償回路 1 6 は、動き補償を行った画素データを、バッファ制御回路 1 2 を介してバッファ 2 のビデオバッファ 2 2 に書き込み、表示出力に備えるとともに、この画素データが、他の画素の参照データとされる場合に備える。

【 0 0 5 7 】

表示出力回路 1 7 は、デコードした画像データを出力するための同期タイミング信号を発生し、このタイミングを基に、バッファ制御回路 1 2 を介して、ビデオバッファ 2 2 から画素データを読み出し、復号ビデオ信号として出力する。

【 0 0 5 8 】

【発明が解決しようとする課題】

以上説明したように、MPEG 2 ストリームは階層構造を有している。図 3 を用いて説明したピクチャ層のsequence\_header乃至picture\_coding\_extensionのデータは、図 2 を用いて説明したプロファイルおよびレベルから成るクラスが異なる場合においても、そのデータ量は、あまり変更されない。一方、スライス層以下のデータ量は、符号化する画素数に依存する。

【 0 0 5 9 】

図 2 より、HLにおいて、1 枚のピクチャで処理しなければならないマクロブロックの数は、MLに対して約 6 倍になる。さらに、図 4 より、4:2:2Pにおいて、1 個のマクロブロックで処理するブロックの数は、MPの 4 / 3 倍になる。

【 0 0 6 0 】

したがって、図 1 5 に示したMP@MLに対応する従来のMPEGビデオデコーダを用

いて4:2:2P@HLの符号化ストリームを復号しようとした場合、VBVバッファサイズおよび画素数の増加に伴って、ストリームバッファ21のバッファサイズが不足する。また、ビットレートの増加に伴い、入力ストリームのストリームバッファ21へのアクセスが増加し、画素数の増加に伴って、動き補償回路16のビデオバッファ22へのアクセスが増加するため、バッファ制御回路12の制御が間に合わなくなる。さらに、ビットレートの増加、マクロブロックおよびブロック数の増加に伴って、デコーダ15の処理が間に合わなくなる。

## 【0061】

一般に、信号処理を高速で実行させようとした場合、回路規模が大幅に増加し、部品点数の増加および消費電力の増加を招いてしまう。したがって、MP@MLを復号する従来技術を用いて、4:2:2P@HLを復号する装置を実現しようとした場合、今日の半導体技術の進展によって信号処理回路、メモリ（バッファ）回路ともに、その動作速度は著しく向上し、その回路規模は縮小化されてはいるものの、実現可能な回路規模で実時間動作が可能な4:2:2P@HLに対応したビデオデコーダを実現することは困難である課題があった。

## 【0062】

ところで、MP@MLの符号化ストリームを、図15に示したMP@MLに対応する従来のMPEGビデオデコーダを用い、1倍速以上で高速再生させることを考えた場合、入力するMP@MLの符号化ストリームから画像を間引いて当該MPEGビデオデコーダに入力する方法が考えられる。

## 【0063】

しかしながら、MPEG方式においては、図1を用いて説明したように、近傍の画像を参照する予測符号化方式を採用しているので、任意の画像を間引いた場合、復号することができない画像が生じてしまう。例えば、図6に示すように、画像 $P_{10}$ を間引いた場合、画像 $P_{10}$ を参照して復号する画像 $B_8$ 、 $B_9$ 、 $B_{11}$ 、 $B_{12}$ を復号することができなくなってしまう。

## 【0064】

そこで、図7に示すように、他の画像に参照されるIピクチャおよびPピクチャは間引かず、他の画像に参照されることがないBピクチャだけを間引いたMP@M

Lの符号化ストリームを入力する方法が考えられる。しかしながら、Bピクチャだけを間引く方法は、Bピクチャだけを検出して間引く機能が必要になる上、Bピクチャだけしか間引かれないので、符号化ストリームの再生速度がIピクチャ、Pピクチャ、およびBピクチャの配置に依存して特定の速度となってしまう課題があった。

【0065】

また、復号したIピクチャ、およびPピクチャを適宜繰り返して表示することにより、見かけ上の再生速度は任意の速度に設定できるものの、そのような表示出力は画像の動きがギクシャクしたものとなってしまう課題があった。

【0066】

本発明はこのような状況に鑑みてなされたものであり、実現可能な回路規模であって、4:2:2P@HLの符号化ストリームを実時間再生でき、且つ、MP@MLの符号化ストリームを高速再生できるビデオデコーダを実現することを目的とする。

【0067】

【課題を解決するための手段】

本発明の第1の復号装置は、高速化された符号化ストリームを入力する入力手段と、高速化された符号化ストリームを復号する複数の復号手段と、複数の復号手段を並行して動作させるように制御する復号制御手段と、複数の復号手段が復号した高速化された符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御手段とを含むことを特徴とする。

【0068】

前記高速化された前記符号化ストリームは、ビットレートが所定数倍に高速化されたMPEG2ビデオビットストリームとすることができる。

【0069】

前記出力制御手段には、ビットレートが所定数倍に高速化されたMPEG2ビデオビットストリームに対応する画像を、0倍乃至所定数倍の再生速度で出力させるようにすることができる。

【0070】

前記復号手段には、復号処理の終了を示す信号を復号制御手段に出力させるよ

うにすることができ、前記復号制御手段には、復号処理の終了を示す信号を出力した復号手段に、新たな符号化ストリームを復号させるように制御させるようにすることができる。

#### 【0071】

本発明の第1の復号装置は、符号化ストリームをバッファリングする第1のバッファ手段と、符号化ストリームから、符号化ストリームに含まれる所定の情報の単位の始まりを表わすスタートコードを読み出すとともに、第1のバッファ手段に、スタートコードが保持されている位置に関する位置情報を読み出す読み出し手段と、読み出し手段により読み出された、スタートコードおよび位置情報をバッファリングする第2のバッファ手段と、第1のバッファ手段による符号化ストリームのバッファリング、および第2のバッファ手段によるスタートコードおよび位置情報のバッファリングを制御するバッファリング制御手段とをさらに含むことができる。

#### 【0072】

本発明の第1の復号装置は、複数の復号手段により復号され、出力された複数の画像データのうちの所定のものを選択する選択手段と、選択手段により選択された画像データの入力を受け、必要に応じて動き補償を施す動き補償手段とをさらに含むことができる。

#### 【0073】

前記復号手段には、復号処理が終了したことを示す終了信号を選択手段に出力させるようにすることができ、前記選択手段は、複数の復号手段のそれぞれの処理状態に対応する値を記憶する記憶手段を含むことができる。さらに、前記選択手段には、前記記憶手段が記憶した値が全て第1の値になった場合、復号処理が終了したことを示す終了信号を出力している復号手段に対応する記憶手段に記憶されている値を、第1の値から第2の値に変更させ、対応する記憶手段に記憶されている値が第2の値である第1の復号手段により復号された画像データうち、いずれかの画像データを選択させ、選択された画像データを復号した復号手段に対応する記憶手段に記憶されている値を第1の値に変更させるようにすることができる。



## 【 0 0 7 4 】

本発明の第 1 の復号装置は、選択手段により選択された画像データ、または動き補償手段により動き補償が施された画像データを保持する保持手段と、選択手段により選択された画像データ、または動き補償手段により動き補償が施された画像データの保持手段による保持を制御する保持制御手段とをさらに含むことができる。

## 【 0 0 7 5 】

前記保持手段には、画像データの輝度成分と色差成分をそれぞれ分けて保持させるようにすることができる。

## 【 0 0 7 6 】

本発明の第 1 の復号方法は、高速化された符号化ストリームを入力する入力ステップと、高速化された符号化ストリームを復号する複数の復号ステップと、複数の復号ステップの処理を並行して動作させるように制御する復号制御ステップと、複数の復号ステップの処理で復号された高速化された符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップとを含むことができる。

## 【 0 0 7 7 】

本発明の第 1 の記録媒体のプログラムは、高速化された符号化ストリームを入力する入力ステップと、高速化された符号化ストリームを復号する複数の復号ステップと、複数の復号ステップの処理を並行して動作させるように制御する復号制御ステップと、複数の復号ステップの処理で復号された高速化された符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップとを含むことができる。

## 【 0 0 7 8 】

本発明の第 2 の復号装置は、高速化された符号化ストリームを入力する入力手段と、複数のスライスデコーダを並行して動作させるように制御するスライスデコーダ制御手段と、複数のスライスデコーダが復号した高速化された符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御手段とを含むことを特徴とする。

## 【 0 0 7 9 】

本発明の第 2 の復号方法は、高速化された前記符号化ストリームを入力する入力ステップと、複数のスライスデコーダを並行して動作させるように制御するスライスデコーダ制御ステップと、複数のスライスデコーダが復号した高速化された符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップとを含むことを特徴とする。

## 【 0 0 8 0 】

本発明の第 2 の記録媒体のプログラムは、高速化された前記符号化ストリームを入力する入力ステップと、複数のスライスデコーダを並行して動作させるように制御するスライスデコーダ制御ステップと、複数のスライスデコーダが復号した高速化された符号化ストリームに対応する画像を任意の再生速度で出力させる出力制御ステップとを含むことを特徴とする。

## 【 0 0 8 1 】

本発明の第 1 の復号装置および方法、並びに記録媒体のプログラムにおいては、高速化された符号化ストリームが入力され、高速化された符号化ストリームが復号される処理が並行して動作されるように制御される。さらに、復号された高速化された符号化ストリームに対応する画像が任意の再生速度で出力される。

## 【 0 0 8 2 】

本発明の第 2 の復号装置および方法、並びに記録媒体のプログラムにおいては、高速化された前記符号化ストリームが入力され、複数のスライスデコーダが並行して動作するように制御される。さらに、複数のスライスデコーダが復号した高速化された符号化ストリームに対応する画像が任意の再生速度で出力される。

## 【 0 0 8 3 】

## 【発明の実施の形態】

本発明を適用した MPEG ビデオデコーダについて、図 1 8 を参照して説明する。図 1 8 は、本発明を適用した MPEG ビデオデコーダ 4 0 の構成例を示している。

## 【 0 0 8 4 】

MPEG ビデオデコーダ 4 0 は、ストリーム入力回路 4 1、スタートコード検出回路 4 2、ストリームバッファ制御回路 4 3、クロック発生回路 4 4、ピクチャデ

コーダ 4 5、スライスデコーダ制御回路 4 6、スライスデコーダ 4 7 乃至 4 9、動き補償回路 5 0、輝度バッファ制御回路 5 1、色差バッファ制御回路 5 2、および表示出力回路 5 3 から構成される IC 3 1、ストリームバッファ 6 1 およびスタートコードバッファ 6 2 で構成され、例えば、DRAM からなるバッファ 6 2、例えば、DRAM からなる輝度バッファ 7 1、例えば、DRAM からなる色差バッファ 7 2、CPU (Central Processing Unit) 3 4、並びに、ドライブ 3 5 で構成される。

## 【 0 0 8 5 】

ストリーム入力回路 4 1 は、高能率符号化された符号化ストリーム (MPEG 2 ビデオビットストリーム) の入力を受け、スタートコード検出回路 4 2 に供給する。スタートコード検出回路 4 2 は、入力された符号化ストリームをストリームバッファ制御回路 4 3 に供給するとともに、図 1 4 を用いて説明したスタートコードを検出して、それを基に、そのスタートコードの種類と、ストリームバッファ 6 1 にそのスタートコードが書き込まれる位置を示す書き込みポインタとを含む、スタートコード情報を生成し、ストリームバッファ制御回路 4 3 に供給する。

## 【 0 0 8 6 】

クロック発生回路 4 4 は、図 1 5 を用いて説明したクロック発生回路 1 3 の 2 倍の周期の基本クロックを発生し、ストリームバッファ制御回路 4 3 に供給する。ストリームバッファ制御回路 4 3 は、クロック発生回路 4 4 から供給される基本クロックに従って、入力された符号化ストリームを、バッファ 3 2 のストリームバッファ 6 1 に書き込み、入力されたスタートコード情報を、バッファ 3 2 のスタートコードバッファ 6 2 に書き込む。ストリームバッファ 6 1 は、少なくとも 4:2:2P@HL のデコードに要求される VBV バッファサイズである 47,185,920 ビットの容量を有している。

## 【 0 0 8 7 】

ピクチャデコーダ 4 5 は、ストリームバッファ制御回路 4 3 を介して、スタートコードバッファ 6 2 からスタートコード情報を読み出す。例えば、デコード開始時は、図 3 を用いて説明した sequence\_header からデコードが開始されるので、ピクチャデコーダ 4 5 は、図 5 を用いて説明したスタートコードである sequence\_header\_code に対応する書き込みポインタをスタートコードバッファ 6 2 から

読み出し、その書き込みポインタを基に、ストリームバッファ 6 1 から `sequence_header` を読み出してデコードする。続いて、ピクチャデコーダ 4 5 は、`sequence_header` の読み出しと同様に、`sequence_extension`、`GOP_header`、`picture_coding_extension` 等をストリームバッファ 6 1 から読み出してデコードする。

#### 【 0 0 8 8 】

ピクチャデコーダ 4 5 が、スタートコードバッファ 6 2 から、最初の `slice_start_code` を読み出した時点で、そのピクチャのデコードに必要な全てのパラメータが揃ったことになる。ピクチャデコーダ 4 5 は、デコードしたピクチャ層のパラメータを、スライスデコーダ制御回路 4 6 に出力する。

#### 【 0 0 8 9 】

スライスデコーダ制御回路 4 6 は、ピクチャ層のパラメータの入力を受け、符号化ストリームのクラス (4:2:2@ML、MP@ML等) を判別する。スライスデコーダ制御回路 4 6 はまた、ピクチャ層のパラメータの入力を受け、ストリームバッファ制御回路 4 3 を介して、スタートコードバッファ 6 2 から、対応するスライスのスタートコード情報を読み出す。さらに、スライスデコーダ制御回路 4 6 は、スライスデコーダ 4 7 乃至 4 9 のいずれかにデコードさせるスライスが、符号化ストリームに含まれる何番目のスライスであるかを示すレジスタを有し、そのレジスタを参照しながら、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスの書き込みポインタをスライスデコーダ 4 7 乃至 4 9 のいずれかに供給する。スライスデコーダ制御回路 4 6 が、スライスデコーダ 4 7 乃至 4 9 のうち、デコードを実行させるスライスデコーダを選択する処理については、図 1 9 および図 2 0 を用いて後述する。

#### 【 0 0 9 0 】

スライスデコーダ 4 7 は、マクロブロック検出回路 8 1、ベクトル復号回路 8 2、逆量子化回路 8 3、および逆 DCT 回路 8 4 で構成され、スライスデコーダ制御回路 4 6 から入力されたスライスの書き込みポインタを基に、対応するスライスを、ストリームバッファ制御回路 4 3 を介してストリームバッファ 6 1 から読み出す。そして、スライスデコーダ制御回路 4 6 から入力されたピクチャ層のパラメータに従って、読み出したスライスをデコードして、動き補償回路 5 0 に出

力する。

【 0 0 9 1 】

マクロブロック検出回路 8 1 は、スライス層のマクロブロックを分離し、各マクロブロックのパラメータをデコードし、可変長符号化された各マクロブロックの予測モードおよび予測ベクトルをベクトル復号回路 8 2 に供給し、可変長符号化された係数データを逆量子化回路 8 3 に供給する。ベクトル復号回路 8 2 は、可変長符号化された、各マクロブロックの予測モードおよび予測ベクトルをデコードして、予測ベクトルを復元する。逆量子化回路 8 3 は、可変長符号化された係数データをデコードして逆DCT回路 8 4 に供給する。逆DCT回路 8 4 は、デコードされた係数データに逆DCTを施し、符号化前の画素データに復元する。

【 0 0 9 2 】

スライスデコーダ 4 7 は、動き補償回路 5 0 に、デコードしたマクロブロックに対する動き補償の実行を要求し（すなわち、図中、REQで示される信号を 1 にする）、動き補償回路 5 0 から動き補償の実行要求に対する受付を示す信号（図中ACKで示される信号）を受けて、デコードされた予測ベクトルおよびデコードされた画素を動き補償回路 5 0 に供給する。スライスデコーダ 4 7 は、ACK信号の入力を受けて、デコードされた予測ベクトルおよびデコードされた画素を動き補償回路 5 0 に供給した後に、REQ信号を 1 から 0 に変更する。そして、次に入力されたマクロブロックのデコードが終了した時点で、REQ信号を、再び 0 から 1 に変更する。

【 0 0 9 3 】

また、スライスデコーダ 4 8 のマクロブロック検出回路 8 5 乃至逆DCT回路 8 8 およびスライスデコーダ 4 9 のマクロブロック検出回路 8 9 乃至逆DCT回路 9 2 においても、スライスデコーダ 4 7 のマクロブロック検出回路 8 1 乃至逆DCT回路 8 4 と同様の処理が行われるので、その説明は省略する。

【 0 0 9 4 】

動き補償回路 5 0 は、スライスデコーダ 4 7 乃至 4 9 から入力されたデータの動き補償が終了したか否かを示すReg\_REQ\_A、Reg\_REQ\_BおよびReg\_REQ\_Cの 3 つのレジスタを有し、これらのレジスタの値を参照しながら、適宜、スライスデコ

ーダ47乃至49のうちの1つを選択して、動き補償実行要求を受け付け（すなわち、REQ信号に対して、ACK信号を出力して、予測ベクトルと画素の入力を受け）、動き補償処理を実行する。このとき、動き補償回路50は、スライスデコーダ47乃至49のうち、所定のタイミングにおいてREQ信号が1であるスライスデコーダ47乃至49に対する動き補償が、それぞれ1回ずつ終了した後に、次の動き補償要求を受け付ける。例えば、スライスデコーダ47が連続して動き補償要求を出しても、スライスデコーダ48およびスライスデコーダ49の動き補償が終了するまで、スライスデコーダ47の2つ目の動き補償要求は受け付けられない。動き補償回路50が、スライスデコーダ47乃至49のいずれのデコーダの出力に対して動き補償を実行するかを選択する処理については、図21および図22を用いて後述する。

## 【0095】

スライスデコーダ47乃至49のいずれかから入力されるマクロブロックが動き補償を使用していない場合、動き補償回路50は、その画素データが輝度データであれば、輝度バッファ制御回路51を介して輝度バッファ71に書き込み、その画素データが色差データであれば、色差バッファ制御回路52を介して色差バッファ72に書き込み、表示出力に備えるとともに、この画素データが、他の画像の参照データとされる場合に備える。

## 【0096】

また、スライスデコーダ47乃至49のいずれかから出力されるマクロブロックが動き補償を使用している場合、動き補償回路50は、スライスデコーダ47乃至49のうち対応するデコーダから入力される予測ベクトルに従って、その画素データが輝度データであれば、輝度バッファ制御回路51を介して、輝度バッファ71から参照画素を読み込み、その画素データが色差データであれば、色差バッファ制御回路52を介して、色差バッファ72から参照画素データを読み込む。そして、動き補償回路50は、読み込んだ参照画素データを、スライスデコーダ47乃至49のいずれかから供給された画素データに加算し、動き補償を行う。

## 【0097】

動き補償回路 5 0 は、動き補償を行った画素データを、その画素データが輝度データであれば、輝度バッファ制御回路 5 1 を介して、輝度バッファ 7 1 に書き込み、その画素データが色差データであれば、色差バッファ制御回路 5 2 を介して、色差バッファ 7 2 に書き込み、表示出力に備えるとともに、この画素データが、他の画素の参照データとされる場合に備える。

## 【 0 0 9 8 】

表示出力回路 5 3 は、デコードした画像データを出力するための同期タイミング信号を発生し、このタイミングに従って、輝度バッファ制御回路 5 1 を介して、輝度バッファ 7 1 から輝度データを読み出し、色差バッファ制御回路 5 2 を介して、色差バッファ 7 2 から色差データを読み出して、復号ビデオ信号として出力する。

## 【 0 0 9 9 】

ドライブ 3 5 は、CPU 3 4 に接続されており、必要に応じて装着される磁気ディスク 1 0 1、光ディスク 1 0 2、光磁気ディスク 1 0 3、および半導体メモリ 1 0 4 などとデータの授受を行う。また、CPU 3 4 は、以上説明した IC 3 1、およびドライブ 3 5 の動作を制御するものである。CPU 3 4 は、例えば、ドライブ 3 5 に装着されている磁気ディスク 1 0 1、光ディスク 1 0 2、光磁気ディスク 1 0 3、および半導体メモリ 1 0 4 などに記録されているプログラムに従って、IC 3 1 に処理を実行させることができる。

## 【 0 1 0 0 】

次に、図 1 9 のフローチャートを参照して、スライスデコーダ制御回路 4 6 の処理について説明する。

## 【 0 1 0 1 】

ステップ S 1 において、スライスデコーダ制御回路 4 6 は、画像の垂直方向のマクロブロック数を設定した後、処理するスライスが画像内の何番目のスライスであるかを表わすレジスタの値 N を 1 に初期化する。ステップ S 2 において、スライスデコーダ制御回路 4 6 は、スライスデコーダ 4 7 が処理中であるか否かを判断する。

## 【 0 1 0 2 】

ステップS2において、スライスデコーダ47が処理中ではないと判断された場合、ステップS3に進む。ステップS3において、スライスデコーダ制御回路46は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスNの書き込みポインタをスライスデコーダ47に供給し、スライスデコーダ47にスライスNをデコードさせ、処理はステップS8に進む。

【0103】

ステップS2において、スライスデコーダ47が処理中であると判断された場合、ステップS4に進む。ステップS4において、スライスデコーダ制御回路46は、スライスデコーダ48が処理中であるか否かを判断する。ステップS4において、スライスデコーダ48が処理中ではないと判断された場合、ステップS5に進む。ステップS5において、スライスデコーダ制御回路46は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスNの書き込みポインタをスライスデコーダ48に供給し、スライスデコーダ48にスライスNをデコードさせ、処理はステップS8に進む。

【0104】

ステップS4において、スライスデコーダ48が処理中であると判断された場合、ステップS6に進む。ステップS6において、スライスデコーダ制御回路46は、スライスデコーダ49が処理中であるか否かを判断する。ステップS6において、スライスデコーダ49が処理中であると判断された場合、ステップS2に戻り、それ以降の処理が繰り返される。

【0105】

ステップS6において、スライスデコーダ49が処理中ではないと判断された場合、ステップS7に進む。ステップS7において、スライスデコーダ制御回路46は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライスNの書き込みポインタをスライスデコーダ49に供給し、スライスデコーダ49にスライスNをデコードさせ、処理はステップS8に進む。

【0106】

ステップS8において、スライスデコーダ制御回路46は、処理するスライスが符号化ストリームの何番目のスライスであるかを示すレジスタの値Nを1だけ



インクリメントする。ステップS 9において、スライスデコーダ制御回路4 6は、全スライスのデコードが終了したか否かを判断する。ステップS 9において、全スライスのデコードが終了されていないと判断された場合、ステップS 2に戻り、それ以降の処理が繰り返される。ステップS 9において、全スライスのデコードが終了されたと判断された場合、スライスデコーダ制御回路4 6の当該処理は終了される。

【0107】

図1 9を用いて説明したスライスデコーダ制御回路4 6の処理の具体例について、図2 0を参照して説明する。上述したように、ピクチャデコーダ4 5でピクチャ層のデータがデコードされ、そのパラメータがスライスデコーダ制御回路4 6に供給される。ここで、図1 9を用いて説明したステップS 1において、スライスデコーダ制御回路4 6は、画像の垂直方向のマクロブロック数（レジスタの値Nが取り得る最大値）を設定した後、レジスタの値Nを1に初期化する。ステップS 2において、スライスデコーダ4 7は処理中ではないと判断されるので、ステップS 3において、スライスデコーダ制御回路4 6は、ピクチャ層のパラメータと、スタートコード情報に含まれるスライス1の書き込みポインタをスライスデコーダ4 7に供給し、スライスデコーダ4 7にスライスN（N=1）をデコードさせ、ステップS 8において、レジスタの値Nを1だけインクリメントする。そして、ステップS 9において、全スライスのデコードが終了していないと判断されるため、処理はステップS 2に戻る。

【0108】

ステップS 2において、スライスデコーダ4 7は処理中であると判断される。そして、ステップS 4において、スライスデコーダ4 8は処理中でないと判断されるので、ステップS 5において、スライスデコーダ制御回路4 6は、ピクチャ層のパラメータと、スライス2の書き込みポインタを、スライスデコーダ4 8に供給し、スライスデコーダ4 8にスライスN（N=2）をデコードさせ、ステップS 8において、Nを1だけインクリメントする。そして、ステップS 9において、全スライスのデコードが終了していないと判断されるため、処理はステップS 2に戻る。

## 【0109】

ステップS2において、スライスデコーダ47は処理中であると判断され、ステップS4において、スライスデコーダ48は処理中であると判断される。そして、ステップS6において、スライスデコーダ49は処理中ではないと判断されるので、ステップS7において、スライスデコーダ制御回路46は、ピクチャ層のパラメータと、スライス3の書き込みポインタを、スライスデコーダ49に供給し、スライスデコーダ49にスライスN (N=3) をデコードさせ、ステップS8において、Nを1だけインクリメントする。そして、ステップS9において、全スライスのデコードが終了していないと判断されるため、処理はステップS2に戻る。

## 【0110】

スライスデコーダ47乃至49は、入力されたスライスのデコード処理を実施した後、デコード処理の完了を示す信号をスライスデコーダ制御回路46に出力する。すなわち、スライスデコーダ47乃至49のいずれかからスライス2のデコードの完了を示す信号が入力されるまで、スライスデコーダ47乃至49は全て処理中であるので、ステップS2、ステップS4、およびステップS6の処理が繰り返される。そして、図20の図中Aで示されるタイミングで、スライスデコーダ48がデコード処理の完了を示す信号を、スライスデコーダ46に出力した場合、ステップS4において、スライスデコーダ48が処理中ではないと判断されるので、ステップS5において、スライスデコーダ制御回路46は、スライス4の書き込みポインタを、スライスデコーダ48に供給し、スライスデコーダ48に、スライスN (N=4) をデコードさせ、ステップS8において、Nを1だけインクリメントする。そして、ステップS9において、全スライスのデコードが終了していないと判断されるため、処理はステップS2に戻る。

## 【0111】

そして、次にスライスデコーダ47乃至49のいずれかからデコード処理の完了を示す信号の入力を受けるまで、スライスデコーダ制御回路46は、ステップS2、ステップS4、およびステップS6の処理を繰り返す。図20においては、スライスデコーダ制御回路46は、図中Bで示されるタイミングで、スライス

デコーダ49からスライス3のデコードの終了を示す信号の入力を受けるので、ステップS6において、スライスデコーダ49は処理中ではないと判断される。ステップS7において、スライスデコーダ制御回路46は、スライス5の書き込みポインタをスライスデコーダ49に供給し、スライスデコーダ49に、スライスN (N=5) をデコードさせ、ステップS8において、Nを1だけインクリメントする。そして、ステップS9において、全スライスのデコードが終了していないと判断されるため、処理はステップS2に戻る。以下、最後のスライスのデコードが終了されるまで、同様の処理が繰り返される。

## 【0112】

このように、スライスデコーダ制御回路46は、スライスデコーダ47乃至49の処理状況に対応してスライスのデコード処理を割り当てるので、スライスデコーダ47乃至49に効率よくデコード処理を実行させることが可能となる。

## 【0113】

次に、動き補償回路50のスライスデコーダ調停処理について、図21のフローチャートを参照して説明する。

## 【0114】

ステップS21において、動き補償回路50は、内部のレジスタReg\_REQ\_A、Reg\_REQ\_BおよびReg\_REQ\_Cを初期化する。すなわち、Reg\_REQ\_A=0、Reg\_REQ\_B=0、Reg\_REQ\_C=0とする。

## 【0115】

ステップS22において、動き補償回路50は、レジスタの値が全て0であるか否かを判断する。ステップS22において、レジスタの値が全て0ではない（すなわち、1つでも1がある）と判断された場合、ステップS24に進む。

## 【0116】

ステップS22において、レジスタの値が全て0であると判断された場合、ステップS23に進む。ステップS23において、動き補償回路50は、スライスデコーダ47乃至49から入力されるREQ信号を基に、レジスタの値を更新する。すなわち、スライスデコーダ47からREQ信号が出力されている場合、Reg\_REQ\_A=1とし、スライスデコーダ48からREQ信号が出力されている場合、Reg\_REQ

\_B=1とし、スライスデコーダ49からREQ信号が出力されている場合、Reg\_REQ\_C=1とする。そして、処理はステップS24に進む。

【0117】

ステップS24において、動き補償回路50は、Reg\_REQ\_A=1であるか否かを判断する。ステップS24において、Reg\_REQ\_A=1であると判断された場合、ステップS25に進む。ステップS25において、動き補償回路50は、スライスデコーダ47にACK信号を送信し、Reg\_REQ\_A=0とする。スライスデコーダ47は、動き補償回路50に、ベクトル復号回路82で復号された予測ベクトルと、逆DCT回路84で逆DCTされた画素を出力する。そして、処理はステップS30に進む。

【0118】

ステップS24において、Reg\_REQ\_A=1ではないと判断された場合、ステップS26に進む。ステップS26において、動き補償回路50は、Reg\_REQ\_B=1であるか否かを判断する。ステップS26において、Reg\_REQ\_B=1であると判断された場合、ステップS27に進む。ステップS27において、動き補償回路50は、スライスデコーダ48にACK信号を送信し、Reg\_REQ\_B=0とする。スライスデコーダ48は、動き補償回路50に、ベクトル復号回路86で復号された予測ベクトルと、逆DCT回路88で逆DCTされた画素を出力する。そして、処理はステップS30に進む。

【0119】

ステップS26において、Reg\_REQ\_B=1ではないと判断された場合、ステップS28に進む。ステップS28において、動き補償回路50は、Reg\_REQ\_C=1であるか否かを判断する。ステップS28において、Reg\_REQ\_C=1ではないと判断された場合、ステップS22に戻り、それ以降の処理が繰り返される。

【0120】

ステップS28において、Reg\_REQ\_C=1であると判断された場合、ステップS29に進む。ステップS29において、動き補償回路50は、スライスデコーダ49にACK信号を送信し、Reg\_REQ\_C=0とする。スライスデコーダ49は、動き補償回路50に、ベクトル復号回路90で復号された予測ベクトルと、逆DCT

回路 9 2 で逆 DCT された画素を出力する。そして、処理はステップ S 3 0 に進む。

#### 【 0 1 2 1 】

ステップ S 3 0 において、動き補償回路 5 0 は、スライスデコーダ 4 7 乃至 4 9 のいずれかから入力されたマクロブロックは、動き補償を使用しているか否かを判断する。

#### 【 0 1 2 2 】

ステップ S 3 0 において、マクロブロックが動き補償を使用していると判断された場合、ステップ S 3 1 に進む。ステップ S 3 1 において、動き補償回路 5 0 は、入力されたマクロブロックに動き補償処理を行う。すなわち、動き補償回路 5 0 は、スライスデコーダ 4 7 乃至 4 9 のうち対応するデコーダから出力される予測ベクトルに従って、その画素データが輝度データであれば、輝度バッファ制御回路 5 1 を介して、輝度バッファ 7 1 から参照画素を読み出し、その画素データが色差データであれば、色差バッファ制御回路 5 2 を介して、色差バッファ 7 2 から参照画素データを読み出す。そして、動き補償回路 5 0 は、読み出した参照画素データを、スライスデコーダ 4 7 乃至 4 9 のいずれかから供給された画素データに加算し、動き補償を行う。

#### 【 0 1 2 3 】

動き補償回路 5 0 は、動き補償を行った画素データを、その画素データが輝度データであれば、輝度バッファ制御回路 5 1 を介して、輝度バッファ 7 1 に書き込み、その画素データが色差データであれば、色差バッファ制御回路 5 2 を介して、色差バッファ 7 2 に書き込み、表示出力に備えるとともに、この画素データが、他の画素の参照データとされる場合に備える。そして、ステップ S 2 2 に戻り、それ以降の処理が繰り返される。

#### 【 0 1 2 4 】

ステップ S 3 0 において、マクロブロックが動き補償を使用していないと判断された場合、ステップ S 3 2 に進む。ステップ S 3 2 において、動き補償回路 5 0 は、その画素データが輝度データであれば、輝度バッファ制御回路 5 1 を介して輝度バッファ 7 1 に書き込み、その画素データが色差データであれば、色差バ

ッファ制御回路 5 2 を介して色差バッファ 7 2 に書き込み、表示出力に備えるとともに、この画素データが、他の画像の参照データとされる場合に備える。そして、ステップ S 2 2 に戻り、それ以降の処理が繰り返される。

#### 【 0 1 2 5 】

図 2 1 を用いて説明した動き補償回路 5 0 によるデコーダの調停処理の具体例について、図 2 2 を参照して説明する。

#### 【 0 1 2 6 】

図 2 2 に示すタイミング C において、図 2 1 のステップ S 2 2 の処理により、動き補償回路 5 0 のレジスタが全て 0 であると判断された場合、スライスデコーダ 4 7 乃至 4 9 は、全て、REQ 信号を出力しているため、ステップ S 2 3 の処理により、それぞれのレジスタの値は、Reg\_REQ\_A = 1、Reg\_REQ\_B = 1、Reg\_REQ\_C = 1 に更新される。そして、ステップ S 2 4 の処理により、Reg\_REQ\_A = 1 であると判断されるため、ステップ S 2 5 において、動き補償回路 5 0 は、スライスデコーダ 4 7 に ACK 信号を出力して、Reg\_REQ\_A = 0 とし、スライスデコーダ 4 7 から予測ベクトルと画素の入力を受け、動き補償 1 を行う。

#### 【 0 1 2 7 】

動き補償 1 が終了した後、すなわち、図 2 2 の D で示されるタイミングにおいて、処理は、再びステップ S 2 2 に戻る。図中 D で示されるタイミングにおいては、スライスデコーダ 4 7 から、REQ 信号が出力されている。しかし、レジスタの値は、Reg\_REQ\_A = 0、Reg\_REQ\_B = 1、Reg\_REQ\_C = 1 であり、ステップ S 2 2 において、レジスタの値は、全て 0 ではないと判断されるため、処理は、ステップ S 2 4 に進み、レジスタの値は更新されない。

#### 【 0 1 2 8 】

ステップ S 2 4 において、Reg\_REQ\_A = 0 であると判断され、ステップ S 2 6 において、Reg\_REQ\_B = 1 であると判断されるので、動き補償回路 5 0 は、ステップ S 2 7 において、スライスデコーダ 4 8 に ACK 信号を出力して、Reg\_REQ\_B = 0 とし、スライスデコーダ 4 8 から予測ベクトルと画素の入力を受け、動き補償 2 を行う。

#### 【 0 1 2 9 】

動き補償 2 が終了した後、すなわち、図 2 2 の E で示されるタイミングにおいて、処理は再びステップ S 2 2 に戻る。図中 E で示されるタイミングにおいても、スライスデコーダ 4 7 から、REQ 信号が出力されている。しかし、レジスタの値は、Reg\_REQ\_A=0、Reg\_REQ\_B=0、Reg\_REQ\_C=1 であるので、ステップ S 2 2 において、レジスタの値は全て 0 ではないと判断されるので、図中 D で示されるタイミングのときと同様、レジスタの値は更新されない。

## 【0 1 3 0】

そして、ステップ S 2 4 において、Reg\_REQ\_A=0 であると判断され、ステップ S 2 6 において、Reg\_REQ\_B=0 であると判断され、ステップ S 2 8 において、Reg\_REQ\_C=1 であると判断されるので、動き補償回路 5 0 は、ステップ S 2 9 において、スライスデコーダ 4 9 に ACK 信号を出力して、Reg\_REQ\_C=0 とし、スライスデコーダ 4 9 から予測ベクトルと画素の入力を受け、動き補償 3 を行う。

## 【0 1 3 1】

動き補償 3 が終了した後、すなわち、図 2 2 の F で示されるタイミングにおいて、処理は再びステップ S 2 2 に戻る。F で示されるタイミングにおいては、レジスタの値は、Reg\_REQ\_A=0、Reg\_REQ\_B=0、Reg\_REQ\_C=0 であるので、ステップ S 2 3 において、レジスタの値が更新され、Reg\_REQ\_A=1、Reg\_REQ\_B=1、Reg\_REQ\_C=0 となる。

## 【0 1 3 2】

そして、ステップ S 2 4 において、Reg\_REQ\_A=1 であると判断され、同様の処理により、動き補償 4 が実行される。

## 【0 1 3 3】

このような処理を繰り返すことにより、動き補償回路 5 0 は、スライスデコーダ 4 7 乃至 4 9 を調停しながら、動き補償を行う。

## 【0 1 3 4】

以上説明したように、本発明を適用した MPEG ビデオデコーダ 4 0 においては、スタートコードバッファ 6 2 を設けたことにより、ピクチャデコーダ 4 5 乃至スライスデコーダ 4 9 を、お互いの動作の終了を待つことなしに、ストリームバッ

ファ61にアクセスさせることができる。また、スライスデコーダ47乃至49は、スライスデコーダ制御回路46の処理により、同時に動作させることができる。さらに、動き補償回路50は、適宜、スライスデコーダ47乃至49のうちの1つを選択し、それぞれ分離された輝度バッファ71および色差バッファ72にアクセスし、動き補償を行うことができる。このように、MPEGビデオデコーダ40は、デコード処理性能およびバッファへのアクセス性能が向上されているので、4:2:2P@HLのMPEG2ビデオビットストリームを実時間でデコードすることが可能となる。

#### 【0135】

ところで、MP@MLのMPEG2ビデオビットストリームを実時間でデコードするために要する処理能力は、4:2:2P@HLのMPEG2ビデオビットストリームを実時間でデコードするために要する処理能力の1/6である。よって換言すれば、本発明を適用したMPEGビデオデコーダ40は、MP@MLのMPEG2ビデオビットストリームを最大6倍速で再生することが可能である。

#### 【0136】

具体的には例えば、図23に示すシステムにMPEGビデオデコーダ40を用いることにより、MP@MLのMPEG2ビデオビットストリーム的高速再生が実現される。

#### 【0137】

当該システムのハードディスク(HDD)111には、MP@MLのMPEG2ビデオビットストリームが記録されている。再生装置112は、MPEGビデオデコーダ40のCPU34からの制御に基づき、ハードディスク111に記録されているMP@MLのMPEG2ビデオビットストリームを、通常(1倍速)よりも高速(例えば6倍速)で読み出し、得られた高速再生ストリームをMPEGビデオデコーダ40に供給する。

#### 【0138】

例えば、図24(A)に示すようなMP@MLのMPEG2ビデオビットストリームは、MPEGビデオデコーダ40により、一切の処理が省略されることなく完全にデコードされ、図24(B)に示すように輝度バッファ71および色差バッファ72に書き込まれる。輝度バッファ71および色差バッファ72に書き込まれた画像データを、表示出力回路53が画像のタイプに拘わりなく6枚当たり1枚の割合



で読み出し、復号ビデオ出力として後段に出力することにより、6倍速の高速再生が実現される。

## 【0139】

なお、表示出力回路53が、輝度バッファ71および色差バッファ72に書き込まれた画像データを、例えば、画像のタイプに拘わりなく3枚当たり1枚の割合で読み出し、復号ビデオ出力として後段に出力すれば、3倍速の高速再生が実現される。

## 【0140】

すなわち、表示出力回路53が、輝度バッファ71および色差バッファ72に書き込まれた画像データを、画像のタイプに拘わりなくX(=0乃至6)枚当たり1枚の割合で読み出し、復号ビデオ出力として後段に出力すれば、X倍速の高速再生が実現される。ここで、X=0の場合、スチル再生となる。

## 【0141】

以上説明したように、MPEGビデオデコーダ40においては、表示出力回路53が、画像のタイプに拘わりなく一定の割合でデータを読み出すことによって、見た目にもギクシャクしていない自然な動きの画像を出力することが可能となる。

## 【0142】

このようにMP@MLのMPEG2ビデオビットストリームを高速再生できるMPEGビデオデコーダ40を映像編集等に用いれば、例えば、ビデオ信号の映像素材の内容を容易に理解することができ、さらに、編集点等を快適に検索することができるので、作業効率を向上させることが可能となる。

## 【0143】

ところでまた、図18に示したMPEGビデオデコーダ40に、さらに表示出力回路と同様の回路を2個追加して表示出力を3系統とすれば、入力された6倍速のMP@MLのMPEG2ビデオビットストリームを、2倍速で再生した復号ビデオ出力を当該3系統で順次出力することが可能となる。

## 【0144】

また、本発明は、2:1インタレースであって30フレーム/秒のMP@MLのMPEG2ビデオビットストリームを2倍速で記録媒体から読み出し、ノンインタレー

スであって60フレーム/秒の画像として出力するような場合にも適用することが可能である。

【0145】

なお、図23に示したシステムにおいては、ハードディスク111にMP@MLのMPEG2ビデオビットストリームを記録するようにしたが、記録されているMP@MLのMPEG2ビデオビットストリームを6倍速で読み出せるものであれば、他の記録媒体を用いてもよい。

【0146】

上述した一連の処理は、ソフトウェアにより実行することもできる。そのソフトウェアは、そのソフトウェアを構成するプログラムが、専用のハードウェアに組み込まれているコンピュータ、または、各種のプログラムをインストールすることで、各種の機能を実行することが可能な、例えば汎用のパーソナルコンピュータなどに、記録媒体からインストールされる。

【0147】

この記録媒体は、図18に示すように、コンピュータとは別に、ユーザにプログラムを提供するために配布される、プログラムが記録されている磁気ディスク101（フロッピーディスクを含む）、光ディスク102（CD-ROM（Compact Disk Read Only Memory）、DVD（Digital Versatile Disk）を含む）、光磁気ディスク103（MD（Mini Disk）を含む）、もしくは半導体メモリ104などよりなるパッケージメディアなどにより構成される。

【0148】

また、本明細書において、記録媒体に記録されるプログラムを記述するステップは、記載された順序に沿って時系列的に行われる処理はもちろん、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理をも含むものである。

【0149】

また、本明細書において、システムとは、複数の装置により構成される装置全体を表すものである。

【0150】

【発明の効果】

以上のように、本発明の第1の復号装置および方法、並びに記録媒体のプログラムによれば、高速化された符号化ストリームを復号する処理を並行して実行するように制御し、復号された高速化された符号化ストリームに対応する画像を任意の再生速度で出力させるようにしたので、実現可能な回路規模であって、MP@MLのMPEG2ビデオビットストリームを任意の再生速度で再生できるビデオデコーダを実現することが可能となる。

【0151】

また、本発明の第2の復号装置および方法、並びに記録媒体のプログラムによれば、高速化された前記符号化ストリームを復号する複数のスライスデコーダを並行して動作するように制御し、復号された高速化された符号化ストリームに対応する画像を任意の再生速度で出力させるようにしたので、実現可能な回路規模であって、MP@MLのMPEG2ビデオビットストリームを任意の再生速度で再生できるビデオデコーダを実現することが可能となる。

【図面の簡単な説明】

【図1】

MPEG方式におけるピクチャタイプを説明するための図である。

【図2】

MPEG2で規定されたプロファイルおよびレベルにおける、各パラメータの上限値を示す図である。

【図3】

MPEG2ビットストリームの階層構造を説明するための図である。

【図4】

マクロブロック層を説明するための図である。

【図5】

sequence\_headerのデータ構造を説明するための図である。

【図6】

sequence\_extensionのデータ構造を説明するための図である。

【図7】

GOP\_headerのデータ構造を説明するための図である。

【図 8】

picture\_headerのデータ構造を説明するための図である。

【図 9】

picture\_coding\_extensionのデータ構造を説明するための図である。

【図 1 0】

picture\_dataのデータ構造を説明するための図である。

【図 1 1】

sliceのデータ構造を説明するための図である。

【図 1 2】

macroblockのデータ構造を説明するための図である。

【図 1 3】

macroblock\_modesのデータ構造を説明するための図である。

【図 1 4】

スタートコードを説明するための図である。

【図 1 5】

MP@MLの符号化ストリームをデコードする従来のMPEGビデオデコーダの構成例を示すブロック図である。

【図 1 6】

符号化ストリームからPピクチャを間引いた場合の問題点について説明するための図である。

【図 1 7】

符号化ストリームからBピクチャだけを間引いた場合の問題点について説明するための図である。

【図 1 8】

本発明を適応したMPEGビデオデコーダ40の構成例を示すブロック図である。

【図 1 9】

スライスデコーダ制御回路46の処理を説明するフローチャートである。

【図 2 0】

スライスデコーダ制御回路 4 6 の処理の具体例を説明するための図である。

【図 2 1】

動き補償回路 5 0 によるスライスデコーダの調停処理を説明するフローチャートである。

【図 2 2】

動き補償回路 5 0 によるスライスデコーダの調停処理具体例を説明するための図である。

【図 2 3】

MPEGビデオデコーダ 4 0 を用いてMP@MLの符号化ストリームを高速再生するシステムの構成例を示すブロック図である。

【図 2 4】

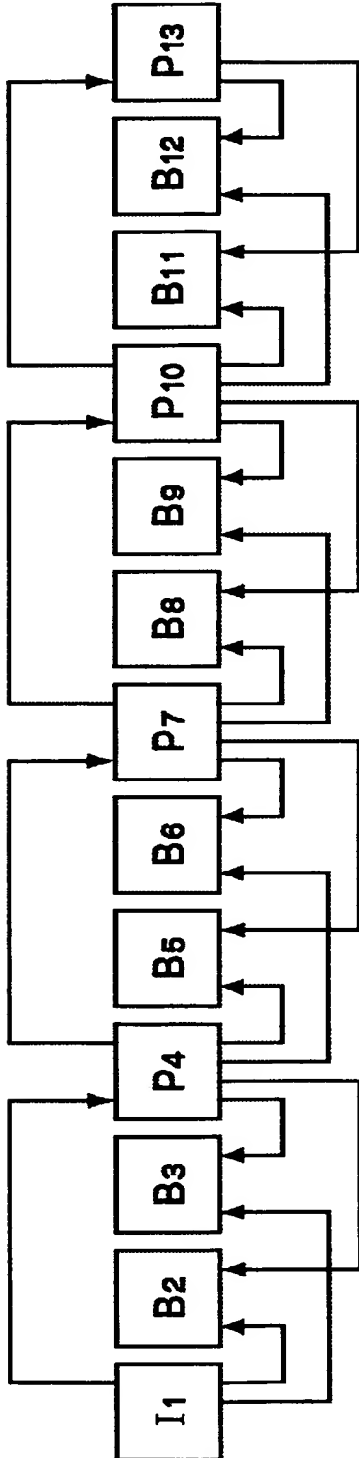
MPEGビデオデコーダ 4 0 がMP@MLの符号化ストリームを高速再生する処理を説明するための図である。

【符号の説明】

3 1 IC, 3 2 バッファ, 3 4 CPU, 4 2 スタートコード検出回路, 4 3 ストリームバッファ制御回路, 4 5 ピクチャデコーダ, 4 6 スライスデコーダ制御回路, 4 7 乃至 4 9 スライスデコーダ, 5 0 動き補償回路, 5 1 輝度バッファ制御回路, 5 2 色差バッファ制御回路, 6 1 ストリームバッファ, 6 2 スタートコードバッファ, 7 1 輝度バッファ, 7 2 色差バッファ, 1 0 1 磁気ディスク, 1 0 2 光ディスク, 1 0 3 光磁気ディスク, 1 0 4 半導体メモリ, 1 1 1 ハードディスク, 1 1 2 再生装置

【書類名】図面

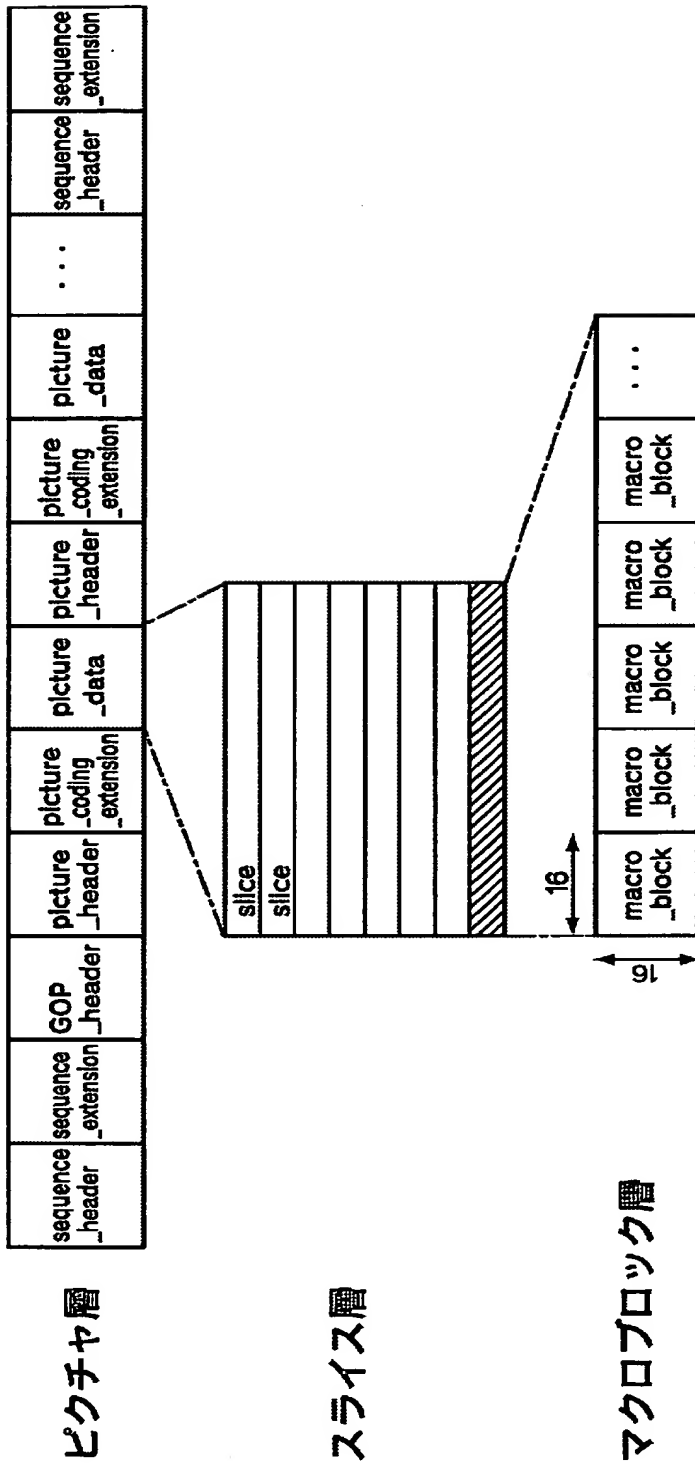
【図 1】



【図 2】

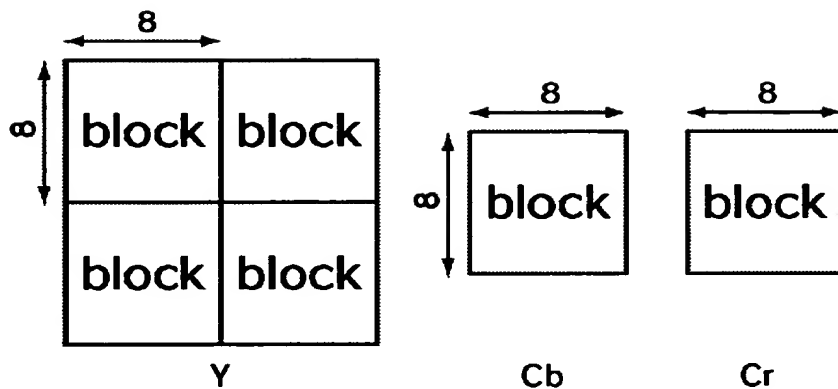
Profile and Level	上限					
	Bit rates(Mbit/s)	Samples/line	Lines/Frame	Frames/sec	Samples/sec	
4:2:2P@HL	300	1920	1152	60	62,668,800	
4:2:2P@ML	50	720	608	30	11,059,200	
MP@HL	80	1920	1152	60	62,668,800	
MP@HL-1440	60	1440	1152	60	47,001,600	
MP@ML	15	720	576	30	10,368,000	
MP@LL	4	352	288	30	3,041,280	
SP@ML	15	720	576	30	10,368,000	

【図 3】



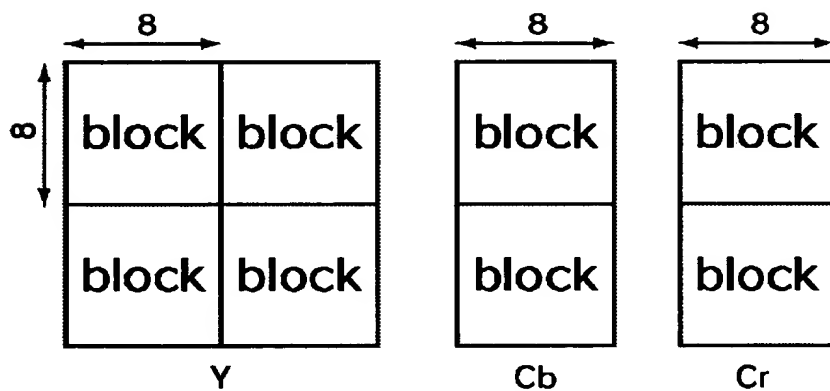


【図 4】



4:2:0方式の macro block層

(A)



4:2:2方式の macro block層

(B)

【図 5】

sequence_header ( ) {	ビット数	二一モニツク
sequence_header_code	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	"1"
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	
load_intra_quantiser_matrix	1	
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code ( )		
}		

sequence\_header

【図 6】

sequence_extension ( ) {	ビット数	ニーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indication	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code ( )		
}		

## sequence\_extension

【図 7】

group_of_picture_header ( ) {	ビット数	ニーモニック
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimsbf
broken_link	1	uimsbf
next_start_code ( )		
}		

## GOP\_header

【図 8】

picture_header ( ) {	ビット数	ニーモニック
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if(picture_coding_type==2    picture_coding_type==3){		
full_pel_forward_vector	1	
forward_f_code	3	uimsbf
}		
if(picture_coding_type==3)		
full_pel_forward_vector	1	
backward_f_code	3	uimsbf
}		
while(nextbits( )=="I") {		
extra_bit_picture/*with the value "I"*/	1	uimsbf
extra_information_picture	8	
}		
extra_bit_picture/*with the value "0"*/	1	uimsbf
next_start_code ( )t		
}		

picture\_header

【図 9】

picture_coding_extension ( ) {	ビット数	ニーモニック
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
f_code[0][0]/*forward horizontal*/	4	uimsbf
f_code[0][1]/*forward vertical*/	4	uimsbf
f_code[1][0]/*backward horizontal*/	4	uimsbf
f_code[1][1]/*backward vertical*/	4	uimsbf
intra_dc_precision	2	uimsbf
picture_structure	2	uimsbf
top_field_first	1	uimsbf
frame_pred_frame_dct	1	uimsbf
concealment_motion_vectors	1	uimsbf
q_scale_type	1	uimsbf
intra_vlc_format	1	uimsbf
alternate_scan	1	uimsbf
repeat_first_field	1	uimsbf
chroma_420_type	1	uimsbf
progressive_frame	1	uimsbf
composite_display_flag	1	uimsbf
if(composite_display_flag) {		
v_axis	1	uimsbf
field_sequence	3	uimsbf
sub_carrier	1	uimsbf
burst_amplitude	7	uimsbf
sub_carrier_phase	8	uimsbf
}		
next_start_code ( )		
}		

picture\_coding\_extension

【図 1 0】

picture_data ( ) {	ビット数	ニーモニック
do {		
slice ( )		
} while(nextbits( )==slice_start_code)		
next_start_code( )		
}		

**picture\_data**

【図 1 1】

slice ( ) {	ビット数	ニーモニック
slice_start_code	32	bslbf
if(vertical_size>2800)		
slice_vertical_position_extention	3	uimsbf
if(<sequence_scalable_extention() is present in the bitstream>)		
if(<scalable_mode=="data partitioning">)		
priority_breakpoint	7	uimsbf
quantiser_scale_code	5	uimsbf
if(nextbits( )=="I") {		
intra_slice_flag	1	uimsbf
intra_slice_	1	uimsbf
reserved_bits	7	uimsbf
while(nextbits( )=="I") {		
extra_bit_slice/*with the value "I"*/	1	uimsbf
extra_information_slice	8	uimsbf
}		
}		
extra_bit_slice/*with the value "0"*/	1	uimsbf
do {		
macroblock ( )		
} while (nextbits( )!='000 0000 0000 0000 0000 0000')		
next_start_code ( )t		
}		

**slice**

【図 1 2】

macroblock( ) {	ビット数	ニーモニック
while (nextbits( )=='0000 0001 000')		
macroblock_escape	11	bslbf
macroblock_address_increment	1-11	vlcbf
macroblock_modes( )		
if(macroblock_quant)		
quantiser_scale_code	5	uimsbf
if(macroblock_motion_forward		
(macroblock_intra && concealment_motion_vectors))		
motion_vectors(0)		
if(macroblock_motion_backward)		
motion_vectors(1)		
if(macroblock_intra && concealment_motion_vectors)		
marker_bit	1	bslbf
if(macroblock_pattern)		
coded_block_pattern( )		
for(1=0; i<block_count; i++) {		
block(i)		
}		
}		

macroblock

【図 1 3】

macroblock_modes( ) {	ビット数	ニーモニック
macroblock_type	1-9	vlcibf
if((spatial_temporal_weight_code_flag==1)&& (spatial_temporal_weight_code_table_index != "00")) {		
spatial_temporal_weight_code	2	uimsbf
}		
if(macroblock_motion_forward    macroblock_motion_backward) {		
if(picture_structure == 'frame') {		
if(frame_pred_frame_dct == 0)		
frame_motion_type	2	uimsbf
} else {		
field_motion_type	2	uimsbf
}		
}		
if((picture_structure == "frame picture") && (frame_pred_frame_dct == 0) && (macroblock_intra1macroblock_pattern)) {		
dct_type	1	uimsbf
}		
}		

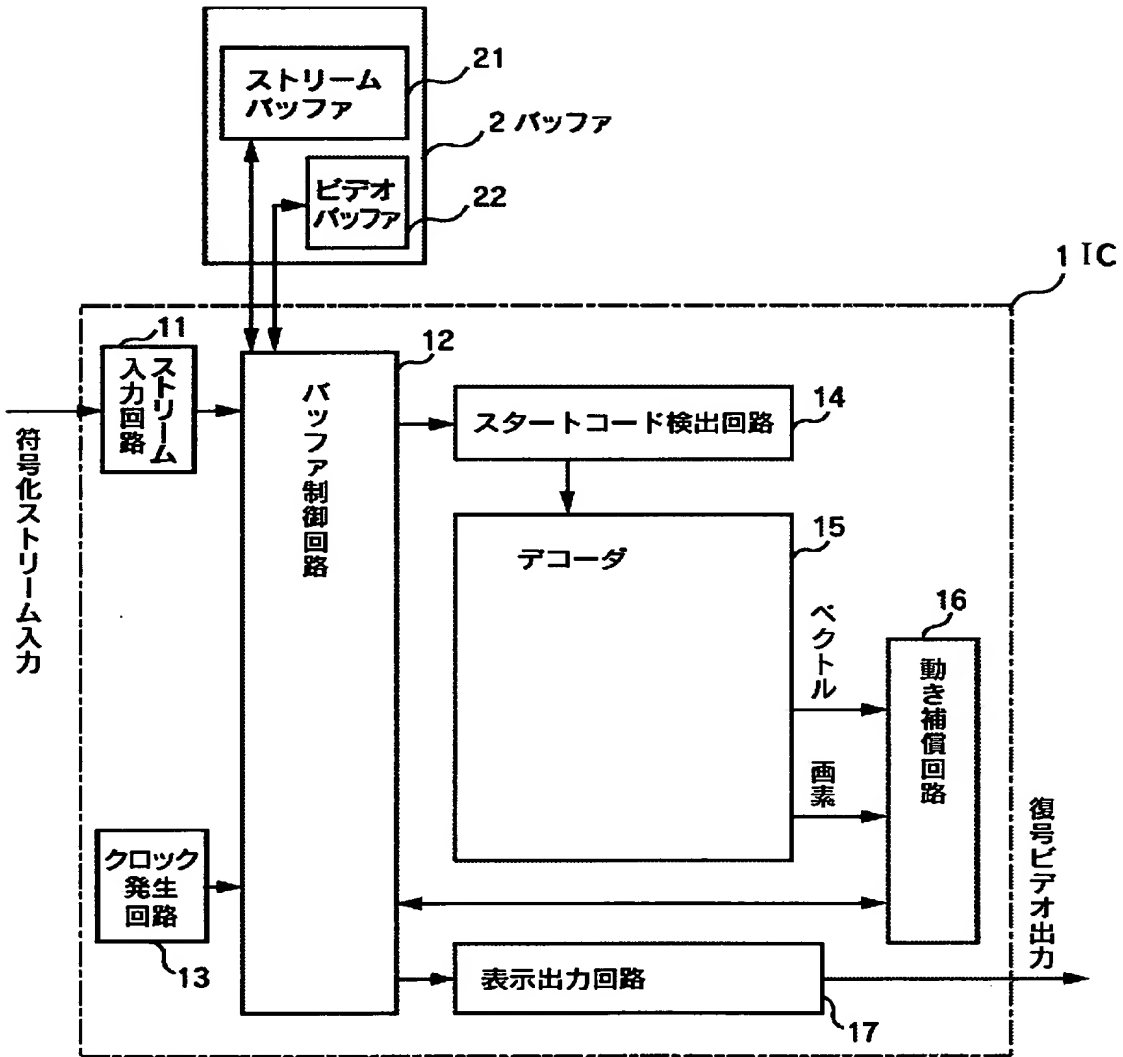
macroblock\_modes



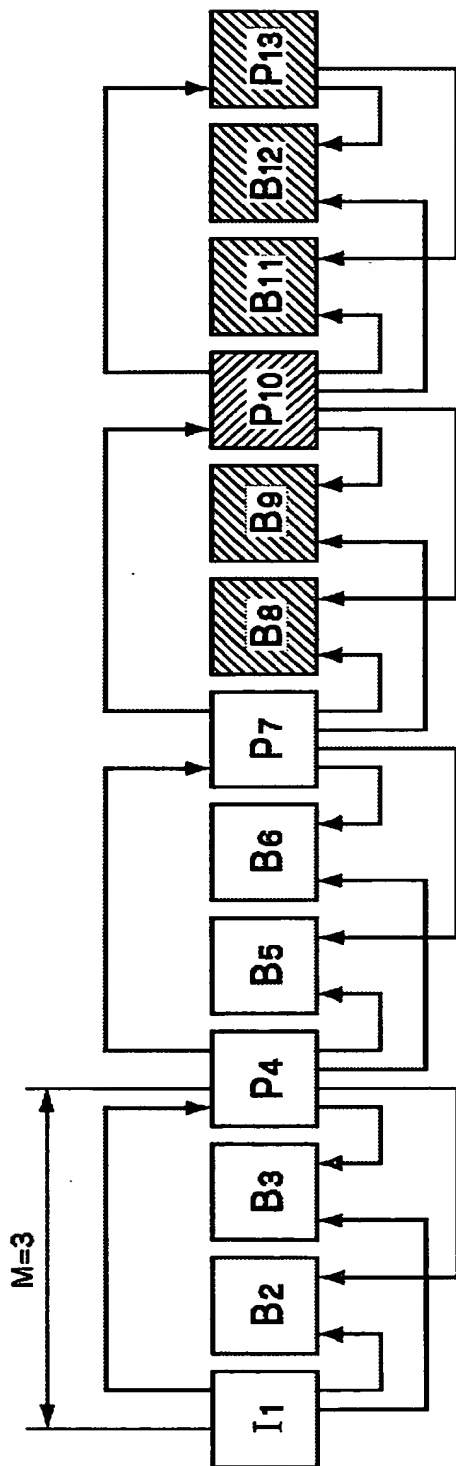
【図 1 4】

名 称	Start code value
Picture_start_code	00
Slice_start_code	01 ~ AF
Reserved	B0
Reserved	B1
User_data_start_code	B2
Sequence_header_code	B3
Sequence_error_code	B4
Extension_start_code	B5
Reserved	B6
Sequence_end_code	B7
Group_start_code	B8
System_start_code	B9 ~ FF

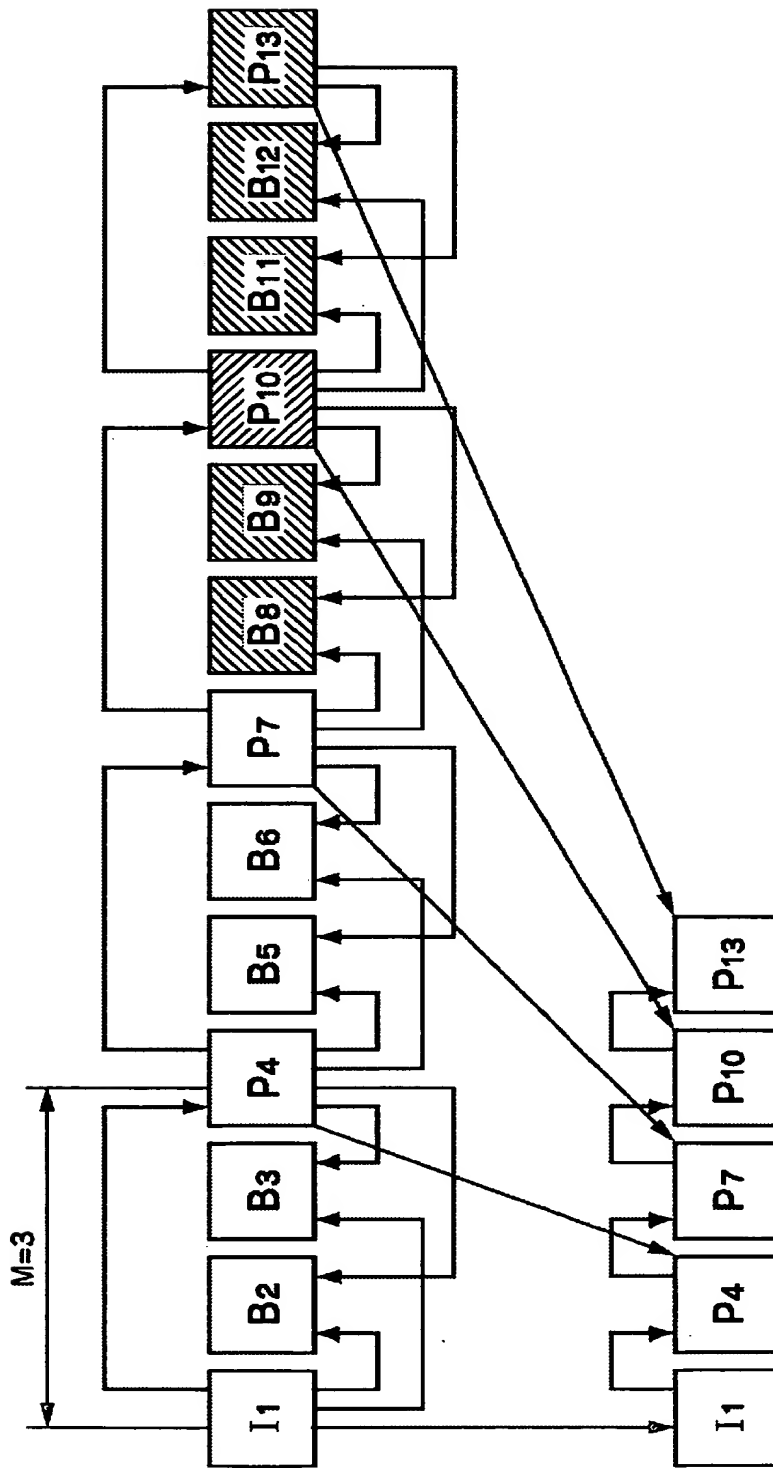
【図 1 5】



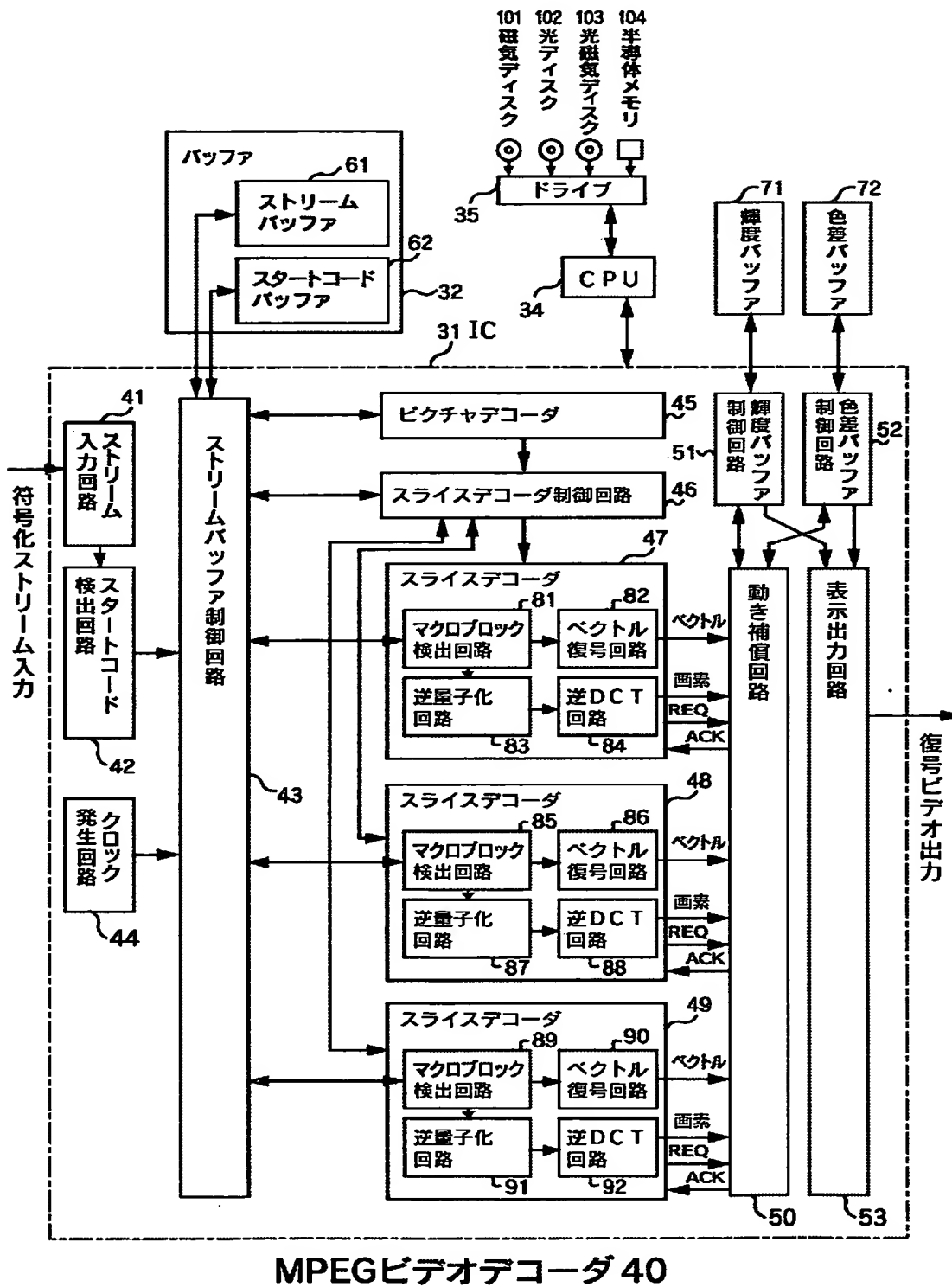
【図 1 6】



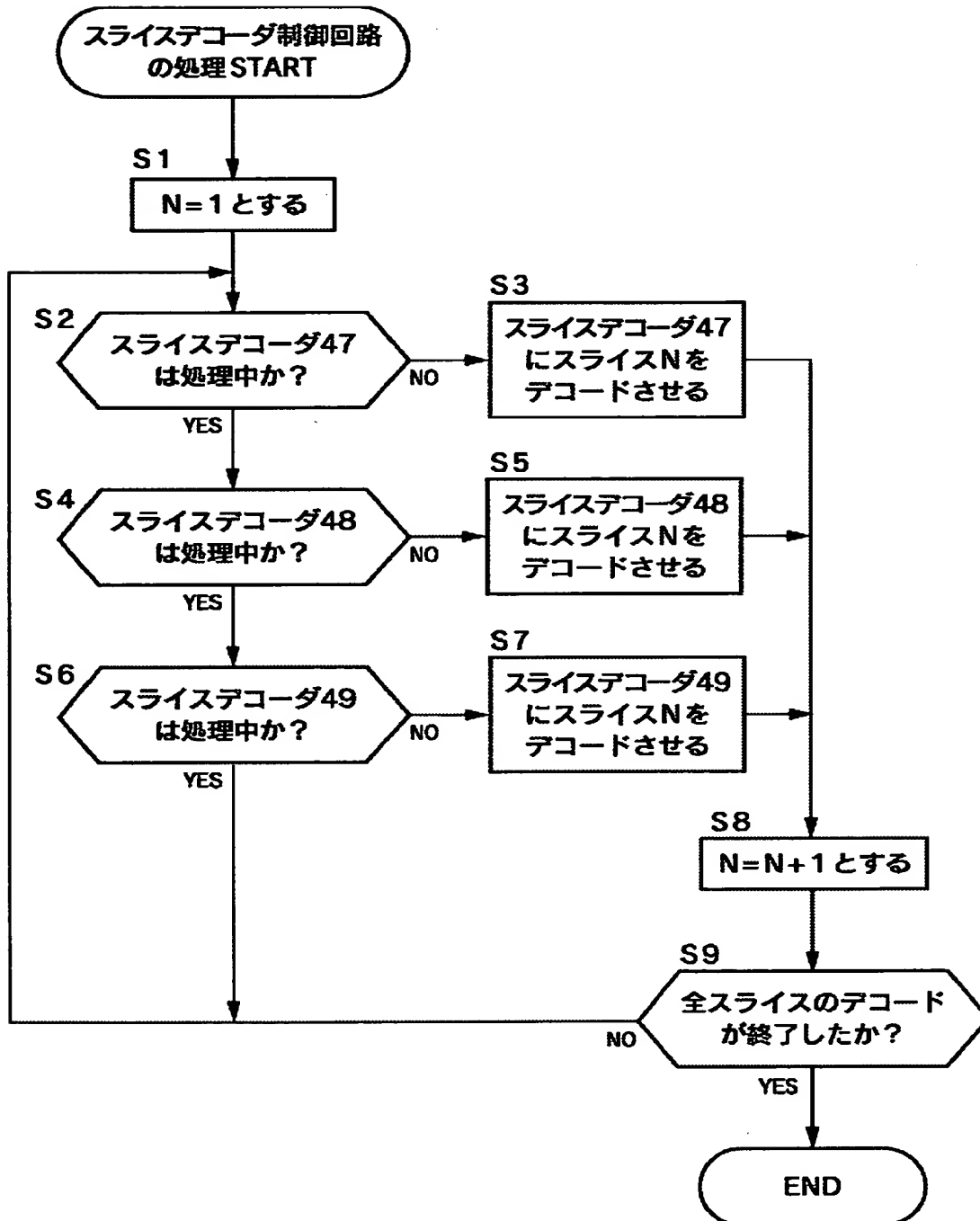
【図 1 7】



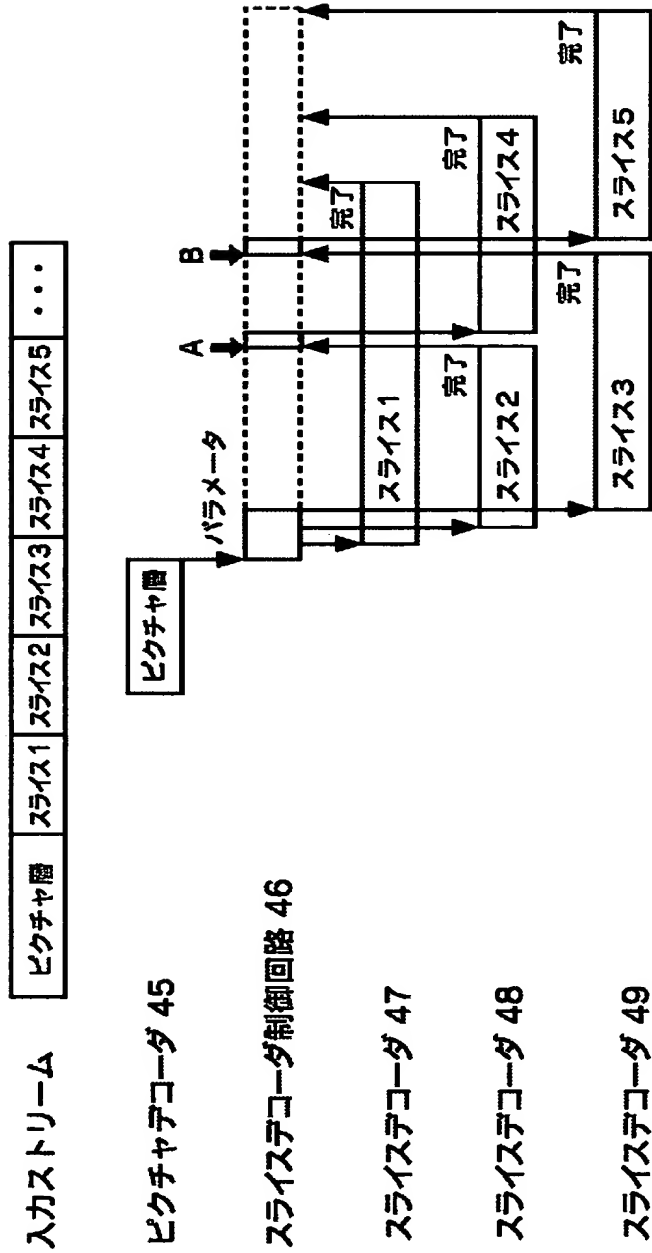
【図 18】



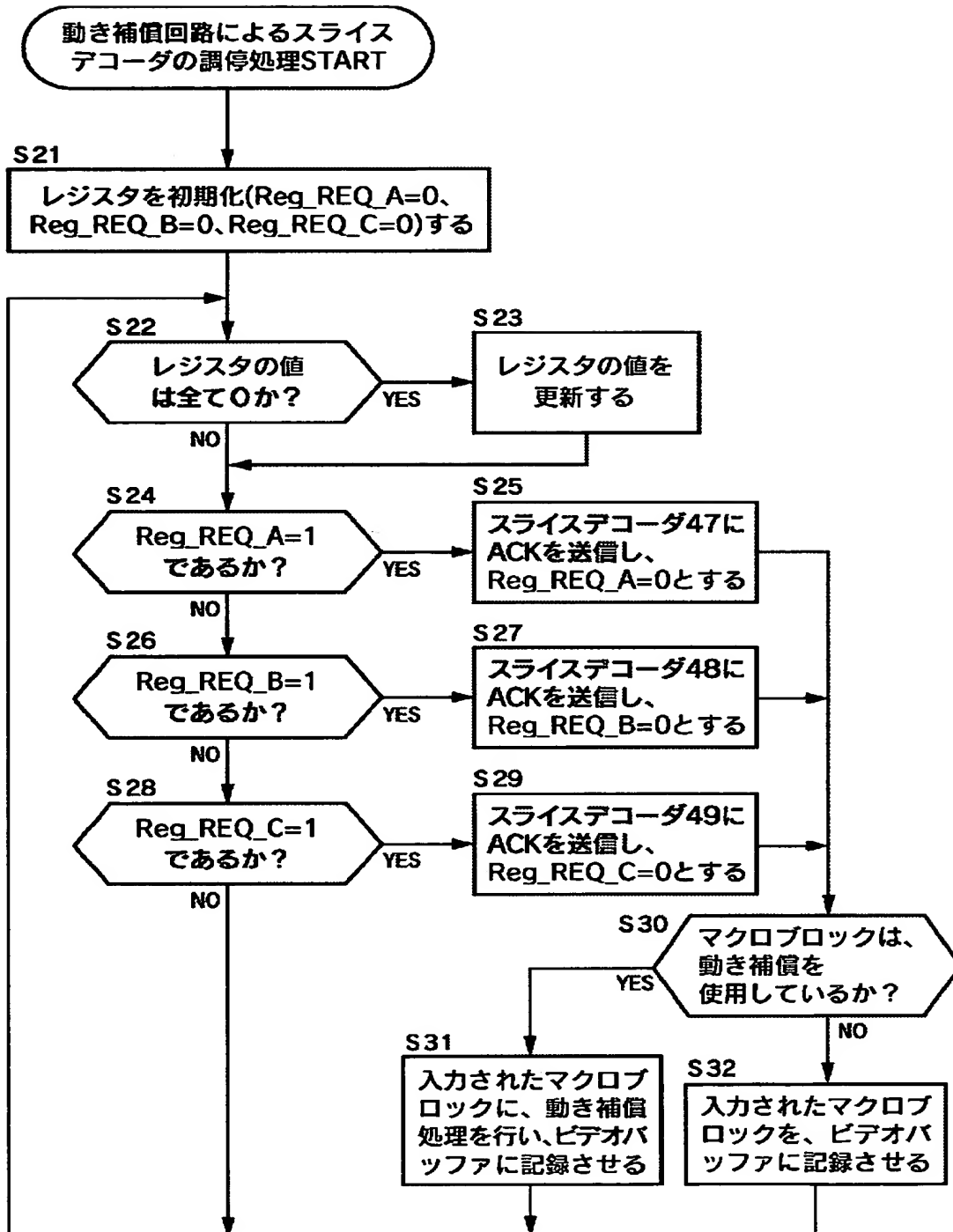
【図 1 9】



【図 20】

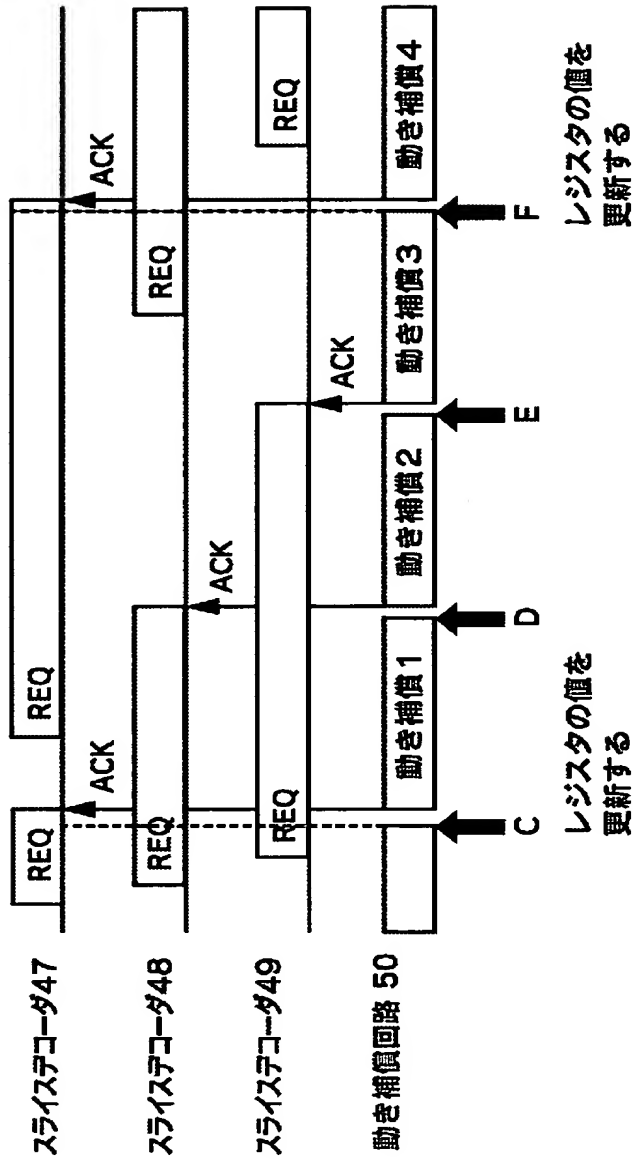


【図 21】

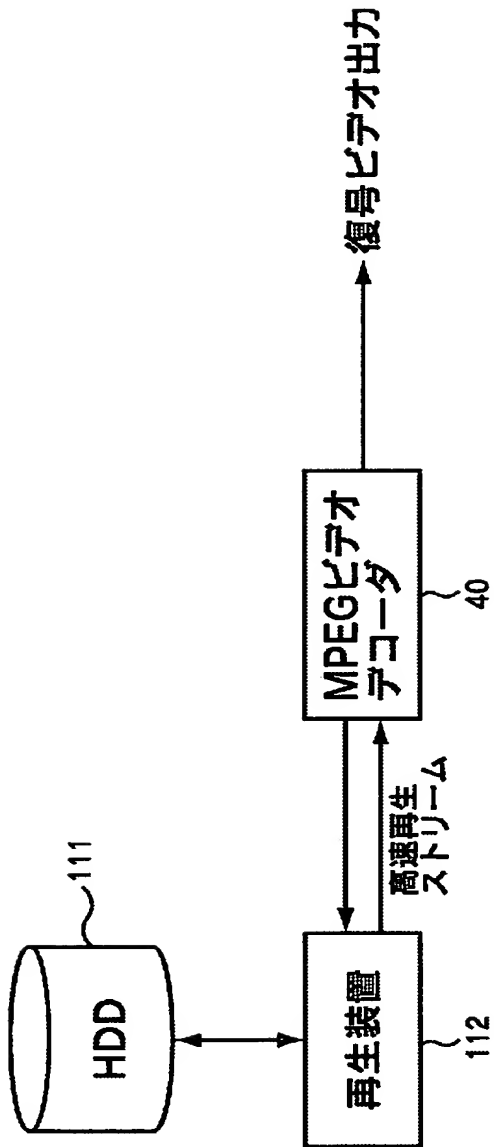




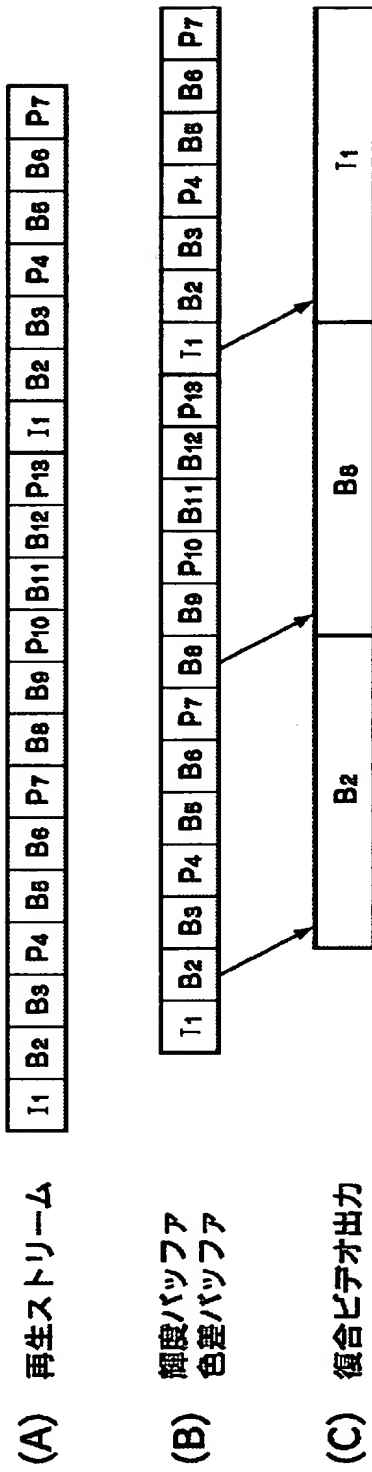
【図 2 2】



【図 2 3】



【図 2 4】



【書類名】 要約書

【要約】

【課題】 MP@MLのMPEG2ビデオビットストリームを任意の速度で再生する。

【解決手段】 パラメータの入力を受けたスライスデコーダ制御回路は、第1のスライスデコーダにピクチャ層のパラメータとスライス1の書き込みポイントを、第2のスライスデコーダにピクチャ層のパラメータとスライス2の書き込みポイントを、第3のスライスデコーダにピクチャ層のパラメータとスライス3の書き込みポイントを、それぞれ順番に供給してデコードさせる。スライスデコーダ制御回路は、第1乃至第3のスライスデコーダから入力されるデコード処理の完了を示す信号の入力に基づいて、第1乃至第3のスライスデコーダにデコード処理を振り分ける。

【選択図】 図20

出 願 人 履 歴 情 報

識別番号 [ 0 0 0 0 0 2 1 8 5 ]

1. 変更年月日 1 9 9 0 年 8 月 3 0 日  
[変更理由] 新規登録  
住 所 東京都品川区北品川 6 丁目 7 番 3 5 号  
氏 名 ソニー株式会社